MIKROSZÁMÍTÓGÉP MAGAZIN
A NEUMANN JÁNOS SZÁMÍTÓGÉP-TUDOMÁNYI TÁRSASÁG LAPJA
Ára: 30 Ft
1986
augusztus
FORMA-1 FOGYASZTÁSBAN 1.
HÍRADÁSTECHNIKA SZOVETKEZET
Sharp
PTA 4000+16

Újdonság!

KODEX 2000 szövegszerkesztő rendszer

Újdonság!

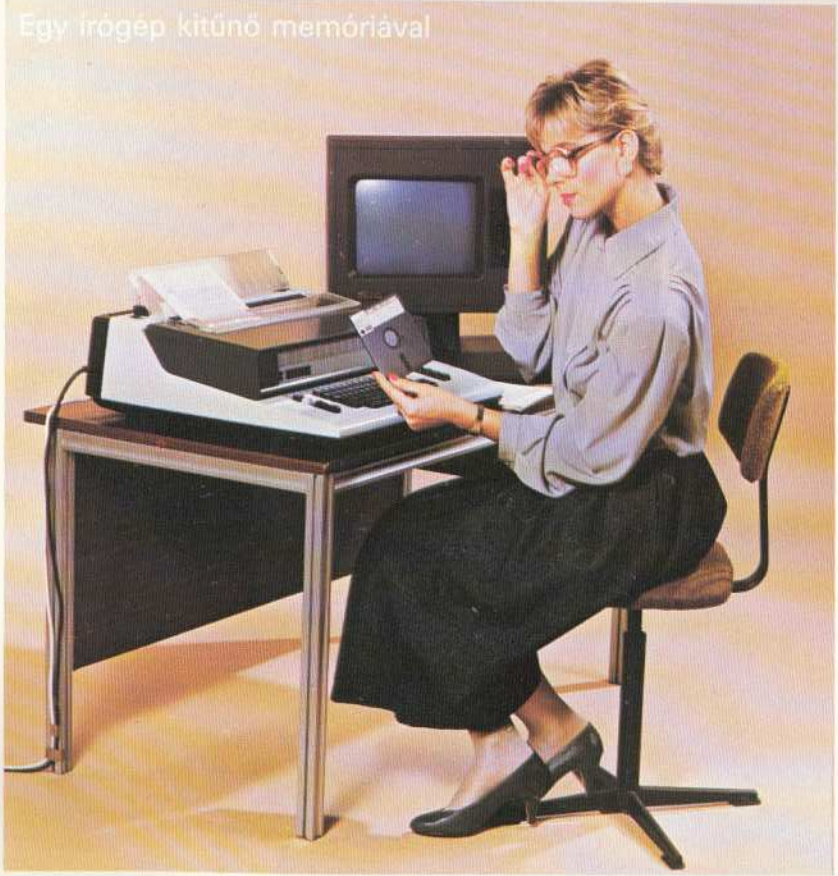
A KODEX 2000 szövegszerkesztő rendszerrel igen könyen, ugyanakkor hatékonyan végezhető szövegek összeállítása és nyomtatása. A program lehetővé teszi, hogy a felhasználó a szöveget videoterminálon, ROBOTRON S 6011 típusú írógépen gépelje le, menetközben változtassa és javitsa, kialakítva a végső formát, majd tetszőleges példányszámban kinyomtassa.

A KODEX 2000 kurzor-orientált rendszer, ami azt jelenti, hogy a felhasználó a szöveg tetszőleges részén dolgozhat, miután a felfelé és lefelé, valamint a balra és jobbra mozgatható kurzort a kivánt pozicióra állította. A KODEX 2000 szövegszerkesztőt dinamikus képernyőformattálás jellemzi, vagyis a képernyő mindig a legutóbbi aktuális állapotot mutatja, a szöveg a nyomtatandó példánnyal azonos alakban jelenik meg. A szövegszerkesztés kezdetekor néhány egyszerű paranccsal megadható a sorhosszúság, a bal margó stb., valamint a fejléc és lap aljának formája. Ezt követően a lap formattálása automatikusan történik. Az aktuális formátumértékek bármikor megjeleníthetők.

A sorhosszon túlnyúló szavak automatikusan a következő sorba kerülnek. A szavakba beiktathatók láthatatlan kötőjelek is, melyek csak akkor kerülnek kinyomtatásra, ha a szó elválasztott formában sor végére kerül.

A KODEX 2000 képernyőjén megjelenő szöveg mozgatható felfelé és lefelé, illetve balra és jobbra. A szöveg vagy egyes különálló sorok középre helyezhetők, vagy balra, ill. jobbra ütköztethetők, a sorok tetszőlegesen ritkíthatók. A szöveg bármely része átmásolható, mozgatható vagy törölhető. A berendezés kezeléséhez számítástechnikai ismeretek nem szükségesek. A KODEX 2000 hatékonyan alkalmazható mindenütt, ahol a fenti lehetőségek előnyt biztositanak a hagyományos gépirási megoldással szemben: pl. szerződések, címlisták, ajánlatok stb. készitése, tárolása, időnkénti karbantartása, automatikus kiiratása, formalevelek készitése, egyszerűbb ügyviteli feladatok elvégzése.

Színvonalas hardver és szoftver ellátás, garancia, folyamatos kellékellátás.



Műszaki adatok:

Írószerkezet: A Robotron 6011 típusú betűtárcsás elektronikus irógép szabvány magyar irógép billentyűzettel carbon- és textilszalag használatának lehetőségével. Esztétikus, kifogástalan iráskép.

Megjelenítő egység (képernyő):
— 25 sor 85 karakter soronként
— teljes magyar karakterkészlet
— folyamatos ernyőtartalom mozgatás (Soft scroll)
— reflexiómentes képcső
— képátló: 12"

Háttér memória:
— 1 db 5 1/4" 2 oldalas, dupla sűrűségű floppy disc egység (a közös házban max. 2 egység helyezhető el)
— 1 Mbyte kapacitás lemezenként

Vezérlő elektronika:
— M 6809 típusú CPU
— 64 kbyte RAM, 32 kbyte EPROM memória (bővíthető max. 128 k RAM-ra)
— 2 soros, 1 párhuzamos interface (opcionális gyorsnyomtató, idegennyelvű billentyűzet csatlakoztatása és lokális hálózat kialakítására)
— floppy csatoló max. 4 lemezegység kiszolgálására
— alfanumerikus display vezérlés

Méret: 480 × 440 × 450mm
Súly: 20 kg
Hálózati feszültség: 220 V, 50 Hz
Teljesítményigény: 120 W
Védettség: kettős szigetelésű
Fejlesztette: BME Folyamatszabályozási Tanszék

Gyártó:
Budapest, XX. ker. H—1201
Helsinki út 53.
Telefon: 279-200
Telex: 22-4399

KONTAKTA

Forgalmazza:
MIGÉRT
Budapest IX. ker. H—1093
Dimitrov tér 14.
Tel.: 175-081
Telex: 22-4736
Író- és számológép osztály

Szerviz:
ITV
Budapest V. ker. H—1052
Bécsi u. 8—10.
Tel.: 184-899
Telex: 22-4381

MIKROSZÁMÍTÓGÉP MAGAZIN

4. ÉVFOLYAM
7. SZÁM
1986. AUGUSZTUS

A NEUMANN JÁNOS SZÁMÍTÓGÉP-TUDOMÁNYI TÁRSASÁG LAPJA

A kiadvány a Tudományszervezési és Informatikai Intézettel együttműködve készül

A szerkesztőbizottság vezetője:
Kovács Győző

E számunkat szerkesztették:
Bakos Tamás (programozástechnika)
Broczkó Péter (hírek)
Kovács Győző (levelezés)
Lindner László (sakkprogramozás)
Petróczy Judit (könyvek)
Simonyi Endre (klub)
Vadkerti János (μprogramok)
Varga András (iskola — számítógép)

A szerkesztőség munkatársa:
Kardos Zsuzsa

PR menedzser
Pálhalmi Vali

Felelős szerkesztő:
Könyves Tóth Pál
Szerkesztőség:
1027 Budapest II., Fő u. 68.
Telefon: 154-250

Kiadja a Delta Szaklapkiadó és Műszaki Szolgáltató Leányvállalat
Felelős kiadó:
dr. Varga György igazgató
1442 Budapest VII., Garay u. 5.
Telefon: 415-583, 215-440

Terjeszti a Magyar Posta
Előfizethető a hírlapkézbesítő hivataloknál
és a Posta Hírlapelőfizetési és Lapellátási Irodáján
(1900 Budapest V., József nádor tér 1.)
vagy átutalással a 215—96162 pénzforgalmi jelzőszámra.
Megjelenik havonta
Egy szám ára 30,— Ft
Előfizetési díj:
egy évre 360,— Ft
fél évre 180,— Ft
Külföldön terjeszti a Kultúra,
1389 Budapest, pf. 149.
és a Magyar Média
1932 Budapest, pf. 279.
86—253

Szikra Lapnyomda
Budapest (86-3475)
Felelős vezető:
Csöndes Zoltán vezérigazgató

INDEX: 25629
ISSN 0236-6088

Címképünk:

A Híradástechnika Szövetkezet PTA—4000-es számítógépe

Tartalom

A nyár és az informatika 2
Assemblerek, cross-assemblerek 11
Hogyan működnek a lapos kijelzők? 14
Adok — veszek — cserélek 15
A minőség közügy 16
Commodore 64 segédtáblázat 23
Olvastunk . . . 36

ISKOLA — SZÁMÍTÓGÉP

Grafikus hardcopy C16-ra 3
Lencsék képalkotása 4
Gépi kódú grafikát segítő program 5
Az ismeretlen C16 6
Mi van a bolgár iskolákban? 7

DIÁKROVAT

A C64 grafikus lehetőségei 8
Kódkonverter 10

PROGRAMOZÁSTECHNIKA

BASIC és gépi kód 17
Z80 programozási gyakorlatok 18

μPROGRAMOK

Disassembler program Sinclair számítógépekre 19
RESTORE n 26
Képernyőn a Primo agya! 27
Megszakításvezéreti BREAK-figyelés 28

μKLUB

PTA—4000 32
PTA—4000. Fedélzeti számítógép az autóban 33

JÁTÉKPROGRAMOK 40

KÖNYVEK 41

AZ OLVASÓ ÍRJA 43

FÓRUM 44

SAKKPROGRAMOZÁS

Bitek és figurák 46

HÍREK, ÉRDEKESSÉGEK 48

A nyár és az informatika

„(Ápám!) A mesterem nem [osztja meg] velem tudását, erős karja [elzár] engem az írnoki tudástól, a tudás határától, hogy kis írnok maradjak: nem [engedi], hogy a tábla házában felsős testvér legyek."

(„A tábla házának fia" — Sumér eredetiből fordította Komoróczi Géza 1970, 1983.)

Talán felesleges is mondanom, hogy a lapkészítés kiszámíthatatlan időbeosztása miatt most május elején éppen júliusban kell gondolkoznom, pedig még csak tegnap kezdődött el a május, még javában virágzanak a fák, a legbátrabbak talán már belekóstoltak a balatoni fürdés örömeibe is. Azért inkább tavasz van még, mint nyár, de a nyár sincs már messze, hiszen egymás után kapom a leveleket, hogy Salgótarjánban, Baján, Kecskeméten, Békéscsabán és ki tudja még merre az országban, szervezik a nyári informatikai táborokat. Megszállott amatőrök, tanárok és ifjúsági vezetők foglalkoznak a tanulni vágyó fiatalokkal, tartanak alapfokú és továbbképző tanfolyamokat, megteremtve az informatikát szerető gyerekeknek azt az idillikus környezetet, amelyben reggeltől estig mással sem foglalkozhatnak, mint a számítógéppel.

Emlékszem az első táborokra vagy 5-6 évvel ezelőtt, amikor különböző manipulációkkal sikerült a táborokba 1-2 gépet szerezni. Ezeket legtöbbször egy gondosan őrzött szobában helyezték el, ahol sokszor zord felügyelők éber tekintete előtt lehetett játszani vagy programot írni, persze csak azoknak, akik a gépekhez értettek.

A fejlődés szerencsére nem állt meg, egyre több ifjúsági táborban jelent meg a számítógép — úgy hallottam, hogy az elmúlt évben már az volt a feltűnő, ha valahol nem lehetett számítógépen dolgozni. Egyket tanárral beszélgettem, akik elmondták, hogy a legtöbb problémát az jelentette, hogy az ifjúság ne csak a számítógép mellett üljön, de focizzon is, ússzon is, kirándduljon is, egyszóval legyen mértékletes a számítógépidő fogyasztásában.

A táborokról szóló hírek az emberben azt a megnyugtató érzést keltik, hogy a diákok nyári informatikai lehetőségei kielégítik az igényeket, ami persze nem azt jelenti, hogy ezután már egyáltalán nem lesznek panaszok, nem lesz egyetlen rosszul szervezett tábor sem, ennyire azért nem lehetünk optimisták.

Ha valamiért szót emelek, annak az az oka, hogy a felnőttek részére — úgy látszik — hasonló lehetőségekről nem gondoskodunk. Pedig az informatika iránti érdeklődés, azt hiszem, egyáltalán nem csökkent, sőt talán még fokozódik is. Ha jól látom a helyzetet, akkor egyre többen látják be, hogy nem elvesztegetett idő, amit a számítástechnika tanulására fordítanak, hiszen egyre több munkahelyen ér többet az, aki a számítógépeket is használni tudja.

Az elmúlt hetekben több, egyetemet és főiskolát, de középiskolát is most végző ifjú ember keresett meg azzal a meglepő hírrel, hogy tanulmányaikat befejezve el szeretnének helyezkedni, de egyre több helyen kérdezik meg tőlük, hogy ismerik-e a számítógépeket, tudnak-e programozni, van-e számítógépes gyakorlatuk. Őszinte örömömre szolgált, hogy — a néhány tíz megkeresésből következtetve — már vannak olyan munkahelyek, ahol az informatikai ismeret a munkát most kezdő szakemberjelölteknél — alapkövetelmény. Horribile dictu! még azt is hallottam, hogy az egyik középvállalatnál a felvétel alkalmából a TV BASIC bizonyítvány határozott előnynek számított, ennek a birtokában ugyanis lelkes telefonálóm többeket megelőzve kapta meg a nem is számítástechnikai állást.

Hozzám forduló barátaim leginkább segítséget kérnek, tanulni szeretnének — olcsón és gyorsan —, pótolni az egyetemeken elmulasztottakat.

A SZÁMALK tanfolyamát szoktam ajánlani, ami az egyéni tanulóknak meglehetősen drága. Úgy hallottam, ezekre a tanfolyamokra főleg a munkahelyek küldik azokat a szakembereket, akiknek az informatikai képzését szükségesnek tartják. A másik probléma, hogy a SZÁMALK-tanfolyamok elsősorban azoknak a szakembereknek szólnak, akik a tanfolyam(ok) elvégzése után főfoglalkozású számítástechnikusként szeretnének a számítástechnika valamelyik részterületével foglalkozni. Ugyanakkor az orvos, a közgazdász, a mérnök vagy bárki más, aki saját szakterületén kíván maradni, de az informatikát is meg akarja ismerni, hogy alkalmazni tudja, nehéz helyzetben van. Ha szerencséjük van, akkor be tudnak iratkozni egy szakmai továbbképző tanfolyamra, ha ott van informatika, vagy maradnak számukra a TIT-tanfolyamok, az NJSZT előadások, esetleg a GM-ek, a PJT-k, vagy ha gépet is vesznek, akkor a gyártó intézmények tanfolyamai.

A következő ötéves tervben a kormány külön forrásokat határozott meg az informatika oktatására, ezen belül az ún. lakossági oktatásra is; azt sajnos még nem tudjuk, hogy mennyit, és azt sem, hogy a forrást mikor lehet igénybevenni. A lakossági oktatásért — jobbnak tűnik, ha informatikai tömegoktatásnak nevezzük — az NJSZT a felelős, de az már biztos, hogy a feladatot egyedül megoldani egyetlen szervezet, így az NJSZT sem tudja. Ezért hoztunk létre széles körű együttműködést mindazokkal a szervezetekkel, amelyeknek nemcsak érdeke, hogy minél több, az informatikához értő ember legyen az országban, de nem keveset tettek is az informatika terjesztése és oktatása érdekében.

Az NJSZT legfontosabb szövetségesei a KISZ, az MTESZ tudományos egyesületek, a TIT, a Hazafias Népfront, a Közművelődési Tanács, hogy csak a legismertebbeket említsem. Hiba lenne az informatikai tömegoktatást önállóan, a közoktatástól és a tanfolyami oktatástól elválasztva kezelni, ezért az oktatási rendszer kialakítása során szorosan együtt kell működnünk a Művelődési Minisztérium intézményeivel, a Tudományszervezési és Informatikai Intézettel, az Országos és a Fővárosi Pedagógiai Intézettel, az Országos Oktatástechnikai Központtal, illetve a tanfolyami oktatásban a Központi Statisztikai Hivatallal és a SZÁMALK-kal, valamint az Ipari Minisztériummal és oktatási szervezeteivel, de sorolhatnánk más főhatóságokat és a hozzájuk tartozó oktatási intézményeket is. Nagyon lényegesnek tartom az OMFB-vel, mint az elektronizációs programért felelős főhatósággal a kapcsolatot, hiszen az informatikai tömegoktatás csak akkor láthatja el jól a feladatát, ha része lesz az elektronizálási programnak.

Látszólag messze kerültem a nyártól és a nyári informatikai táboroktól, valójában nem. Én azt hiszem, hogy a felnőttek részére szervezett informatikai tömegoktatás hatékonyságát is növelni lehetne oly módon, ahogyan ma az ifjúsági informatikai táborokat szervezzük. Sőt — meg sem kellene várni, amíg a rendszeres tömegoktatás beindul, a nyaralást már most is fel lehetne használni alapfokú informatikai ismeretek megszerzésére. Egészen biztos vagyok abban, hogy sokan szívesen áldoznák szabadidejüket, nyaralásuk egy részét arra, hogy a nyaralásuk alatt az informatika alkalmazásával megismerkedjenek. Arra gondolok, hogy az üdülési szezonban — általában a szervezett üdülés 2-2 hétig tart — az üdülők egy részében kéthetes informatikai tanfolyamot lehetne szervezni, amelyhez az előadást a televízióban és a rádióban, a számítógépeket pedig a vállalati, illetve a SZOT-üdülőkben lehetne biztosítani. Ma, amikor egy számítógép kb. tízezer forintba kerül, hozzá a legegyszerűbb tv legfeljebb négyezer, azt hiszem, hogy a vállalatoknak megérné, hogy üdülő munkatársaiknak a fele vagy a negyede úgy jöjjön vissza a nyári szabadságról, hogy tud már valamit a számítógépről és a számítástechnikáról. Nagyon sokszor ér az a vád, hogy megvalósíthatatlan ötleteim vannak, már hallom is, naivitás azt hinnem, hogy majd a számítógép lép az ultiparti vagy a römizés helyébe. Én ezt nem hiszem, de azt igen, hogy a szokásos szórakozás mellett üdítő változatosság lehet az informatika, ez az intellektuális játék, ami nemcsak hasznos ismereteket ad, de unaloműzőnek sem akármilyen szórakozás!

Amikor a μM-nak ezt a számát olvassák, már bőven tart az üdülési szezon, egy ilyen televíziós-rádiós nyári továbbképző tanfolyam megszervezésére most már nagyon kevés az idő. De arra talán lenne lehetőség, hogy a nem túlságosan kihasznált személyi számítógépeiket a vállalatok nyárra az üdülőkbe telepítsék, és néhány szünidős diáknak azért fizessenek, hogy a tanulni vágyó dolgozóknak elmagyarázzák, mit is lehet a számítógépekkel csinálni.

Nyár van. Ne felejtsük el, az informatika korában a nyár a pihenve tanulás ideje!

KOVÁCS GYŐZŐ

Grafikus hardcopy C16-ra

A C16 népszerűségét többek között jó grafikájának is köszönheti, ami BASIC-ből is könnyen elérhető.

Programom egy viszonylag rövid (1 page) gépi rutin, amellyel a grafikus képernyő tartalma menthető ki nyomtatóra. Keletkezésének története szorosan összefügg az Úry-féle C16-könyv 236. oldalával. Itt található ugyanis az a BASIC program, amelynek ugyanez lenne a célja, de sajnos ez nem működik. Lehet, hogy nyomdai hiba, bár nem valószínű. Így hát készítettem egy másik programot, de már teljes gépi kódban, szubrutinként hívhatóan.

A &HCOPY monitorból olvasható be, egyszerűen az L paranccsal. Belépési címe &1700, BASIC-ből, a SYS DEC ("1700") utasítással hívható.

Ha jobban szemügyre vesszük a programot, látható, hogy nem is igazán &1700-nál, hanem &179F-nél kezdődik. Itt kezdődik a Kernal-rutinok felhasználásával a nyomtató megnyitása (kb. OPEN 4,4). Ezután következik a program lelke, illetve annak meghívása a JSR &1710-zel.

A program megírásában az első nehézséget az jelentette, hogy a nyomtató a grafikus adatokat csak függőleges formájában tudja feldolgozni. Ezzel szemben a képernyőpontok a memóriában 8-as „csomagokban" tárolódnak. Egy 8×8-as karakternyi bitrácsot forgat át a &171F-től kezdődő pár sor. Ez a &1708-tól kezdve 8 bájton elhelyezi a karakter felső hét sorát, de úgy, hogy ez már nyomtatásra alkalmas legyen, vagyis egy bájtba helyezve az azonos biteket, és mindenütt b7 = 1. Az így elkészített adatsort a &160F-en levő rutin küldi el a nyomtatónak.

Az előbb pár sorról beszéltem, mégpedig azért, mert a &171F rutin a nyomtató egy másik furcsaságát is leküzdi, hogy csak hét pontsort tudok egyszerre kiírni. Itt lép be &DE eltolásjelző, ami megmutatja, hogy mennyi maradt meg az előző sorban vett nyolcas karaktercsomagból. Ugyancsak itt használom a &DC—&DD, valamint a &171D—&171E-n levő mutatókat.

Ennek a „hétsorosságnak" egy másik következménye, hogy a képernyő hasznos területéből az utolsó kiírandó adatsorba csak 4 bitsor maradt. Ezért kellett a &160F rutinba egy AND-et elhelyezni.

A program a kiírás után lezárja a nyomtatót (&17F1) és RET-tel tér vissza.

TASSONYI KADOCSA

```
16F0  A2 08     LDX #$08
16F2  BD 07 17  LDA $1707,X
16F5  2D 06 17  AND $1706
16F8  20 D2 FF  JSR $FFD2
16FB  CA        DEX
16FC  D0 F4     BNE $16F2
16FE  60        RTS
16FF  00        BRK
1700  4C 9F 17  JMP $179F
1703  4C 00 00  JMP $0000
1706  8F        ???
1707  FF        ???
1708  BF        ???
1709  A8        TAY
170A  C8        INY
170B  98        TYA
170C  88        DEY
170D  F8        SED
170E  98        TYA
170F  88        DEY
1710  A4 DE     LDY $DE
1712  A5 DC     LDA $DC
1714  8D 1D 17  STA $171D
1717  A5 DD     LDA $DD
1719  8D 1E 17  STA $171E
171C  B9 78 40  LDA $4078,Y
171F  A2 08     LDX #$08
1721  2A        ROL
1722  7E 07 17  ROR $1707,X
1725  CA        DEX
1726  D0 F9     BNE $1721
1728  C8        INY
1729  C8        INY
172A  98        TYA
172B  88        DEY
172C  29 07     AND #$07
172E  C5 DE     CMP $DE
1730  F0 1A     BEQ $174C
1732  C0 08     CPY #$08
1734  D0 E6     BNE $171C
1736  A0 00     LDY #$00
1738  AD 1D 17  LDA $171D
173B  18        CLC
173C  69 40     ADC #$40
173E  8D 1D 17  STA $171D
1741  AD 1E 17  LDA $171E
1744  69 01     ADC #$01
1746  8D 1E 17  STA $171E
1749  18        CLC
174A  90 D0     BCC $171C
174C  A2 08     LDX #$08
174E  A9 FF     LDA #$FF
1750  2A        ROL
1751  7E 07 17  ROR $1707,X
1754  CA        DEX
1755  D0 F9     BNE $1750
1757  20 F0 16  JSR $16F0
175A  EA        NOP
175B  C6 DF     DEC $DF
175D  A5 DF     LDA $DF
175F  D0 16     BNE $1777
1761  A9 0D     LDA #$0D
1763  20 D2 FF  JSR $FFD2
1766  A5 DE     LDA $DE
1768  D0 F5     BNE $175F
176A  A5 DC     LDA $DC
176C  38        SEC
176D  E9 40     SBC #$40
176F  85 DC     STA $DC
1771  A5 DD     LDA $DD
1773  E9 01     SBC #$01
1775  85 DD     STA $DD
1777  A5 DC     LDA $DC
1779  18        CLC
177A  69 08     ADC #$08
177C  85 DC     STA $DC
177E  A5 DD     LDA $DD
1780  69 00     ADC #$00
1782  85 DD     STA $DD
1784  A5 DF     LDA $DF
1786  D0 16     BNE $179E
1788  A9 28     LDA #$28
178A  85 DF     STA $DF
178C  C6 DE     DEC $DE
178E  A9 07     LDA #$07
1790  25 DE     AND $DE
1792  85 DE     STA $DE
1794  A9 08     LDA #$08
1796  20 D2 FF  JSR $FFD2
1799  CE 07 17  DEC $1707
179C  A9 00     LDA #$00
179E  60        RTS
179F  A2 04     LDX #$04
17A1  A0 00     LDY #$00
17A3  A9 04     LDA #$04
17A5  20 BA FF  JSR $FFBA
17A8  20 C0 FF  JSR $FFC0
17AB  A2 04     LDX #$04
17AD  20 C9 FF  JSR $FFC9
17B0  A9 0D     LDA #$0D
17B2  20 D2 FF  JSR $FFD2
17B5  A9 08     LDA #$08
17B7  20 D2 FF  JSR $FFD2
17BA  A9 00     LDA #$00
17BC  85 DC     STA $DC
17BE  85 DE     STA $DE
17C0  A9 20     LDA #$20
17C2  85 DD     STA $DD
17C4  A9 28     LDA #$28
17C6  85 DF     STA $DF
17C8  A9 1C     LDA #$1C
17CA  8D 07 17  STA $1707
17CD  A9 FF     LDA #$FF
17CF  8D 06 17  STA $1706
17D2  20 10 17  JSR $1710
17D5  AD 07 17  LDA $1707
17D8  D0 F3     BNE $17CD
17DA  A9 8F     LDA #$8F
17DC  8D 06 17  STA $1706
17DF  20 10 17  JSR $1710
17E2  AD 07 17  LDA $1707
17E5  10 F8     BPL $17DF
17E7  A9 0F     LDA #$0F
17E9  20 D2 FF  JSR $FFD2
17EC  A9 0D     LDA #$0D
17EE  20 D2 FF  JSR $FFD2
17F1  A9 04     LDA #$04
17F3  20 C3 FF  JSR $FFC3
17F6  20 CC FF  JSR $FFCC
17F9  60        RTS
17FA  31 37     AND ($37),Y
17FC  30 30     BMI $182E
17FE  04        ???
```

Lencsék képalkotása

A program C16-os gépre készült; felhasználható az általános iskolák 8. osztályos fizikaóráin a fénytan tanításánál. Domború és homorú lencsék képalkotását mutatja be. A fókusztávolság és a képtávolság megadása után a képernyőn megjelenik a kép szerkesztéséhez szükséges sugármenet, a tárgytávolság és a nagyítás számértéke.

A program a TII által forgalmazott, ékezetes betűket tartalmazó gépre készült, ezért a listában a kiírandó szövegek kissé „furcsák"; a nyomtató ugyanis az eredetileg definiált karaktereket írja ki. A program begépelésekor ezeket a sorokat a következőképpen írjuk:

2 PRINTTAB (11) "LENCSÉK KÉPALKOTÁSA"
4 SCNCLR:PRINT "Figyelj a fókusztávolság megadásánál!"
5 PRINT:PRINT "A homorú lencse fókusztávolsága [r] negatív [R] !"
7 SCNCLR:INPUT "A lencse fókusztávolsága (cm)"; g:INPUT "A tárgytávolság (cm)"; t
2Ø PRINT "[r] A képtávolság" . . .
21 PRINT "[r] A nagyítás" . . .
22 PRINT "Új adatok → [r] U [R]
Befejezés → [r] B [R] ";
35 . . . PRINT "[r] Nincs képalkotás" . . .

A program begépelésénél ügyeljünk arra, hogy szóközt ne nagyon írjunk, mert akkor esetleg a program nem fér be a finom grafika miatt szűkös memóriaterületre.

A program ötletet adhat általános iskolai tanároknak tantárgyi programok írására.

DR. PORONYI GÁBOR

```
1 GRAPHIC2,1:PRINTCHR$(14):COLOR0,2:COLOR1,1
2 PRINTTAB(11)"L?/?♥+/ /+?L/T! ?♥♠"
3 FORI=1TO4000:NEXT
4 SCNCLR:PRINT"-IGYELJ A F ?KUSZT/VOLS/G MEGAD/S/N/L!"
5 PRINT:PRINT"♠ HOMOR? LENCSE F ?KUSZT/VOLS/GA ?NEGATIV?!"
6 FORK=1 TO 5000:NEXT
7 SCNCLR:INPUT"♠ LENCSE F ?KUSZT/VOLS/GA (CM)";G:INPUT"♠ T/RGYT/VOLS/G (CM)";T
8 IFT<=0ORG=0THEN7:ELSEF=ABS(G)
9 DRAW1,0,80TO320,80:FORK=60TO260STEP50:DRAW1,K,77TOK,83:NEXT
10 IFG>0THENGOSUB24:ELSEGOSUB25:GOTO15
11 IFT>2*FTHENP=150:GOSUB27:GOSUB29:GOTO19
12 IFT=2*FTHENP=100:GOSUB27:GOSUB31:GOTO19
13 IFT>FTHENP=75:GOSUB27:GOSUB33:GOTO19
14 IFT=FTHENP=50:GOSUB27:GOSUB35:GOTO22:ELSEP=30:GOSUB27:GOSUB36:GOTO19
15 IFT>2*FTHENP=150:GOSUB27:GOSUB40:GOTO19
16 IFT=2*FTHENP=100:GOSUB27:GOSUB42:GOTO19
17 IFT>FTHENP=75:GOSUB27:GOSUB44:GOTO19
18 IFT=FTHENP=50:GOSUB27:GOSUB46:ELSEP=30:GOSUB27:GOSUB47
19 K=T*G/(T-G)
20 PRINT"?♠ K?PT/VOLS/G: ";:PRINTUSING"####.##";K;:PRINT" CM "
21 PRINT"?♠ NAGY?T/S: ";:PRINTUSING"###.##";ABS(K/T)
22 PRINT"    ?J ADATOK -> ?U?    ?EFEJEZ/S -> ?B?";
23 GETKEYA$:IFA$="U"THEN7:ELSEIFA$="B"THENGRAPHIC0:END:ELSE23
24 CIRCLE1,120,80,52,,50,130:CIRCLE1,200,80,52,,230,310:RETURN
25 CIRCLE1,95,80,62,,50,130:CIRCLE1,225,80,62,,230,310
26 DRAW1,142,40TO178,40:DRAW1,142,120TO178,120:RETURN
27 DRAW1,158-P,80TO158-P,50TO162-P,50TO162-P,80:PAINT1,161-P,77
28 DRAW1,158-P,50TO160,50:RETURN
29 DRAW1,160,50TO245,101:DRAW1,10,50TO245,97
30 DRAW1,234,80TO234,95TO236,95TO236,80:PAINT1,235,81:RETURN
31 DRAW1,160,50TO270,116:DRAW1,60,50TO270,113
32 DRAW1,258,80TO258,110TO262,110TO262,80:PAINT1,259,81:RETURN
33 DRAW1,160,50TO320,148:DRAW1,85,50TO320,145
34 DRAW1,309,80TO309,142TO316,142TO316,80:PAINT1,310,81:RETURN
35 DRAW1,160,50TO310,140:DRAW1,110,50TO310,170:PRINT"?/INCS K?PALKOT/S":RETURN
36 DRAW1,160,50TO220,86:DRAW1,130,50TO220,140
37 L=50:FORK=130TO80STEP-2:DRAW1,K,L:L=L-2:NEXT
38 L=50:FORK=160TO80STEP-3:DRAW1,K,L:L=L-1.5:NEXT:DRAW1,97,80TO97,21TO104,21TO10
4,80
39 RETURN
40 GOSUB48:DRAW1,10,50TO210,90
41 DRAW1,122,80TO122,72:DRAW1,125,80TO125,72:RETURN
42 GOSUB48:DRAW1,60,50TO210,95
43 DRAW1,128,80TO128,70:DRAW1,125,80TO125,70:RETURN
44 GOSUB48:DRAW1,85,50TO210,100:DRAW1,128,80TO128,67:DRAW1,131,80TO131,67:RETURN
45 RETURN
46 GOSUB48:DRAW1,110,50TO210,110:DRAW1,134,80TO124,66TO127,66TO137,80:RETURN
47 GOSUB48:DRAW1,130,50TO210,130:DRAW1,140,80TO140,62TO143,62TO142,80:RETURN
48 DRAW1,160,50TO210,20:L=50:FORK=160TO110STEP-8:DRAW1,K,L:L=L+1.9:NEXT:RETURN
```

Kiegészítés
a „Pályázat közoktatási számítógépes programok elkészítésére" c. felhíváshoz (Mikroszámítógép Magazin, 1986. április)

A Tudományszervezési és Informatikai Intézet értesíti az érdeklődőket,
hogy a Közoktatási Programpályázaton
az új iskolaszámítógépre írt programokkal is részt lehet venni.
Jelenleg a következő gépeken futó programokkal lehet pályázni.

1. Általános iskolai kategóriában: COMMODORE—16, PRIMO, PRO/PRIMO, TV-COMPUTER

2. Középiskolai kategóriában: COMMODORE—64, HT (valamennyi típus), PRIMO, PRO/PRIMO, TV-COMPUTER

A pályázat részletes feltételeinek leírását — kérésre — postán megküldjük vagy személyesen átadjuk.

Címünk: Tudományszervezési és Informatikai Intézet,

Budapest XI., Egry József utca 1—9. — BME „E" épület — XI. em. 5.

PÁRIS GYÖRGY sk.
A Tudományszervezési és Informatikai Intézet igazgatója

HT–1080Z Gépi kódú grafikát segítő program

A BASIC-ben készíthető grafika minden számítógépnél, így a HT-nál is lassú. Ez a lassúság elsősorban olyan esetekben, amikor ugyanabban a programban egy-egy képernyőképet változatlan vagy csak kevésbé módosított formában többször meg kell jeleníteni (játékprogramok, demonstrációs programok stb.), igen zavaró lehet, és egy magára „valamit adó" programozó számára nem is kielégítő megoldás.

A képek gépi kódban történő elkészítése a sebességi problémákat megoldja ugyan, de körülményes, lassú és idegőrlő munka.

A program, melyet itt bemutatok, a képernyőképeket BASIC-ből állítja elő, majd a kész képet a BASIC számára el nem érhető, a RAMTOP feletti tárterületre helyezi át *(1. program)*. Az áthelyezett képek a forgalomban levő számos segédprogram egyikével, például a MONIT-tal betölthető gépi kódú ("SYSTEM") programmá másolhatók ki a tárból. Célszerű a képeket tároló tármezőket és az ismertetésre kerülő segédprogramot egymás mellé illeszteni; így egyetlen, összefüggő gépi kódú programhoz jutunk.

Az így előállított programot a RAMTOP feletti mezőbe betöltjük, majd BASIC betöltéssel bevisszük a BASIC programot is. Ezután a tárban levő képek a BASIC-ből hívott blokkmozgató gépi kódú rutinnal igen gyorsan átvihetők a képernyőre.

1. program

```
 10 CLS:DEFSTR B
 20 B1=CHR$(191)+CHR$(128):B2=CHR$(128)+CHR$(191)
 30 B4=CHR$(128):B5=CHR$(191)
 40 GOSUB120:GOSUB260
 50 GET3C00:PUT7000:FIELD0400:SAVE
 60 CLS:B3=B1:B1=B2:B2=B3:B6=B4:B4=B5:B5=B6
 70 GOSUB120:GOSUB260
 80 PUT7400:SAVE
 90 PUT7400:LOAD:FORI=0TO70:NEXT
100 PUT7000:LOAD:FORI=0TO70:NEXT
110 GOTO90
120 REM KERET
130 REM KERET FELSO
140 A=0:B=B1:BB=B5:GOSUB400
150 A=64:B=B2:BB=B4:GOSUB400
160 REM KERET FUGGOLEGES
170 FORI=128TO704STEP128
180 PRINT@I,B1;:PRINT@(I+61),B2;:PRINT@(I+64),B2;
190 PRINT@(I+64),B2;:PRINT@(I+125),B1;
200 NEXT
210 PRINT@I,B1;:PRINT@(I+61),B2;
220 REM KERET ALSO
230 A=832:B=B2:BB=B4:GOSUB400
240 A=896:B=B1:BB=B5:GOSUB400
250 RETURN
260 REM "GRAPH"
270 DATA150,169,128,151,169,128,150,169
280 DATA128,151,169,128,149,170
290 DATA149,176,128,181,154,128,181,186
300 DATA128,181,154,128,181,186
310 DATA165,186,128,151,164,128,149,170
320 DATA128,149,128,128,149,170
330 FORJ=344TO477STEP64
340 FORI=0TO13:READA:PRINT@(J+I),CHR$(A);:NEXTI
350 NEXTJ
360 RESTORE
370 PRINT@592,"GEPI KODU GRAFIKAT SEGITO PROGRAM";
380 PRINT@662,"KESZITETTE: NAGY IMRE";
390 RETURN
400 REM KERET VIZSZINTES RESZE
410 FORI=0TO60STEP2:PRINT@(A+I),B;:NEXT
420 PRINTBB;
430 RETURN
```

2. program

```
                        ;GEPI KODU GRAFIKA SEGITESE
                        ;
                        ;
                        ;A BASIC BOVITES
                                ORG   417DH
 6 417D C3807F                  JP    FIELD
 7 4180 C3689F                  JP    GET
 8 4183 C3907F                  JP    PUT
 9                              ORG   4188H
10 4188 C3A87F                  JP    ALOAD
11                              ORG   41A0H
12 41A0 C3987F                  JP    SAVE
13                              ORG   41E2H
14 41E2 E9                      JP    (HL)    ;AUTOM. INDULAS
15                      ;
16                      ;FOPROGRAM
17                              ORG   7F80H
18 7F80 CDBE7F  FIELD:  CALL  ADDR    ;A BYTE-OK SZAMA
19 7F83 ED53B87F        LD    (BYTS),DE
20 7F87 C9              RET
21 7F88 CDBE7F  GET:    CALL  ADDR    ;A VIDEO KEZDOCIM
22 7F8B ED53BA7F        LD    (SCRN),DE
23 7F8F C9              RET
24 7F90 CDBE7F  PUT:    CALL  ADDR    ;A TAR KEZDOCIM
25 7F93 ED53BC7F        LD    (MEMADR),DE
26 7F97 C9              RET
27 7F98 E5      SAVE:   PUSH  HL      ;VIDEO ---> TAR
28 7F99 2ABA7F          LD    HL,(SCRN)
29 7F9C ED5BBC7F        LD    DE,(MEMADR)
30 7FA0 ED4BB87F        LD    BC,(BYTS)
31 7FA4 EDB0            LDIR
32 7FA6 E1              POP   HL
33 7FA7 C9              RET
34 7FA8 E5      ALOAD:  PUSH  HL      ;TAR ---> VIDEO
35 7FA9 2ABC7F          LD    HL,(MEMADR)
36 7FAC ED5BBA7F        LD    DE,(SCRN)
37 7FB0 ED4BB87F        LD    BC,(BYTS)
38 7FB4 EDB0            LDIR
39 7FB6 E1              POP   HL
40 7FB7 C9              RET
41                      ;
42                      ;A BYTE-OK SZAMA ES A CIMEK
43 7FB8 0000    BYTS:   DW    0H
44 7FBA 0000    SCRN:   DW    0H
45 7FBC 0000    MEMADR: DW    0H
46                      ;
47                      ;A CIMEK TOMORITESE 2 BYTE-BA
48 7FBE CDE57F  ADDR:   CALL  CHARIN
49 7FC1 CB27            SLA   A       ;BITEK A 4 MSB HELYRE
50 7FC3 CB27            SLA   A
51 7FC5 CB27            SLA   A
52 7FC7 CB27            SLA   A
53 7FC9 57              LD    D,A
54 7FCA 23              INC   HL      ;A SZOVEGMUT.NOVELESE
55 7FCB CDE57F          CALL  CHARIN
56 7FCE B2              OR    D
57 7FCF 57              LD    D,A     ;"H" BYTE KESZ
58 7FD0 23              INC   HL      ;A SZOVEGMUT.NOVELESE
59 7FD1 CDE57F          CALL  CHARIN
60 7FD4 CB27            SLA   A       ;BITEK A 4 MSB HELYRE
61 7FD6 CB27            SLA   A
62 7FD8 CB27            SLA   A
63 7FDA CB27            SLA   A
64 7FDC 5F              LD    E,A
65 7FDD 23              INC   HL      ;A SZOVEGMUT.NOVELESE
66 7FDE CDE57F          CALL  CHARIN
67 7FE1 B3              OR    E
68 7FE2 5F              LD    E,A     ;"L" BYTE KESZ
69 7FE3 23              INC   HL      ;A SZOVEGMUT.NOVELESE
70 7FE4 C9              RET
71 7FE5 AF      CHARIN: XOR   A       ;CIMKARAKTER BE
72 7FE6 86              ADD   A,(HL)
73 7FE7 FE40            CP    40H     ;BETU?
74 7FE9 DAEE7F          JP    C,NUM   ;NEM
75 7FEC D637            SUB   37H     ;IGEN
76 7FEE E60F    NUM:    AND   0FH     ;MASZKOLAS 4 LSB-RE
77 7FF0 C9              RET
78                      END
```

A program a HT—BASIC lemezes rendszerének utasításait használja, így elkerülhetők az USR alkalmazásával járó kényelmetlenségek.

A felhasznált utasítások és formátumuk:

PUTnnnn	a tármező kezdőcímének megadása
GETnnnn	a VIDEO-RAM mező kezdőcímének megadása
FIELDnnnn	a mozgatni kívánt bájtok száma
SAVE	átvitel a képernyőről a tárba
LOAD	átvitel a tárból a képernyőre

A GET, a PUT és a FIELD utasításokban a címek (nnnn) hexadecimálisak. A bemutatott formátum kötelező, az utasításokban felesleges karakterek (például betűköz) nem lehetnek, és a címeknek mindig 4 karakterből kell állniuk.

A program 16 kbájtos gépre készült. 64 kbájtos gép alkalmazásakor a FOPROGRAM megjegyzést követő ORG 7F80H helyett ORG ØFF80H-t kell írni, így a program ebben az esetben is a tár végére kerül.

A programot az EDI assembler-editorral kell elkészíteni. A hibátlan forrásprogramot az A paranccsal és OPTION C-vel mindjárt kazettára fordítjuk, az EXEC.ADDR kérdésre Ø6CCH-t adva meg. A programnév tetszőleges lehet. A gépi kódú program betöltés után automatikusan elindul, a vezérlést a BASIC-nek visszaadva. Mivel a 41E2H című rekeszben a JP (HL) utasítás kódja van, a SYSTEM parancs használatakor előre ki nem számítható problémák adódhatnak. Célszerű ezért ennek a rekesznek a tartalmát a BASIC program kezdetén eredeti értékére visszaállítani egy POKE 16866,195 utasítással.

Ha a program alkalmazása során (például csak egy képet kívánunk mozgatni) a GET, PUT és FIELD utasításokkal nem változtatjuk meg a tármutatókat és a bájtszámlálót, ezek az utasítások a LOAD alkalmazása előtt feleslegesek.

A program alkalmazásának szemléltetésére bemutatok egy példát. A *2. program* BASIC-ben előállítja az *ábrán* látható képet, ezt tárolja, előállítja a „keret" inverzét, ezt is tárolja, majd a két képet váltakozva jeleníti meg a képernyőn.

NAGY IMRE

Az ismeretlen C16

Rendszerváltozók, beépített rutinok

1985 decemberében az ÁPISZ jóvoltából hozzájutottam egy C16-os mikroszámítógéphez. Mivel a hozzáférhető irodalomból viszonylag kevés információt kaptam, magam láttam a gép lelkivilágának feltárásához. Bár ez a munka még nem fejeződött be, szeretném tapasztalataimat megosztani az ez iránt érdeklődőkkel. E cikksorozat keretében először a gép rendszerváltozóiról és beépített rutinjairól írok.

Az egyes változók, eljárások leírásánál igyekszem egységesen eljárni. A megnevezéssel kezdem, amely általában egy angol nyelvű azonosító és megegyezik a C64-leírásokban található azonosítókkal. Ezután az azonosítóhoz tartozó cím következik, előbb hexadecimálisan, majd zárójelben decimálisan. Ez a cím változók esetén a változó kezdő- és végcíme, eljárások esetén pedig az eljárás belépési pontja.

Ezután rövid leírás következik a használatról, működésről. Végül, kapcsolódva a „BASIC és gépi kód" c. cikksorozathoz, a változó vagy eljárás címe a C64 és VC20-as gépeken, csak hexadecimálisan, esetleg annak jelzése, hogy a szóban forgó eljárás vagy változó ezeken a gépeken nem található meg.

A hibakezelésről annyit, hogy a C64-en és a VC20-on hiba esetén a programfutás hibaüzenettel félbeszakad. A C16-on ez rendszerint csak akkor történik meg, ha saját hibakezelő rutinunk nem aktív (lásd a TRAP BASIC utasítást).

Változók

LINNUM $14-$15 (20—21)
16 bites, előjel nélküli egész szám. A BASIC általában utasássorszámok, illetve gépi címek átmeneti tárolására használja. (C64, VC20: ugyanitt)

TXTTAB $2B—$2C (43—44)
Mutató a BASIC programterület és a programszöveg kezdetére. (C64, VC20: ugyanitt)

VARTAB $2D—$2E (45—46)
Mutató a BASIC változóterület kezdetére. (C64, VC20: ugyanitt)

ARYTAB $2F—$30 (47—48)
Mutató a BASIC tömbök kezdetére. (C64, VC20: ugyanitt)

STREND $31—$32 (49—50)
Mutató a BASIC tömbök utáni első bájtra. Ez a cím a szabad terület alsó határa. (C64, VC20: ugyanitt)

FRETOP $33—$34 (51—52)
Mutató a karakterláncok által elfoglalt terület alsó határára. Ez egyúttal a szabad terület utáni első, nem szabad bájt címe. (C64, VC20: ugyanitt)

FRESPC $35—$36 (53—54)
Segédmutató a karakterlánc-műveletek elvégzéséhez. A létrehozandó karakterlánc kijelölt kezdőcímét tartalmazza. A programozási gyakorlatban nem találkozunk vele, mert csak a BASIC rendszer belső rutinjai használják. Talán ez az oka annak, hogy az irodalomban sokszor „megfeledkeznek" ennek a változónak a létezéséről. (C64, VC20: ugyanitt)

MEMSIZ $37—$38 (55—56)
A BASIC által használható RAM-terület felső határának mutatója. (C64, VC20: ugyanitt)

TXTPTR $3B—$3C (59—60)
BASIC szövegmutató: a feldolgozás alatti BASIC programszöveg aktuális karaktercímét mutat. Elsősorban a CHRGET eljárás használja. (C64, VC20: $7A—$7B)

SAREG $07F2 (2034)
A SYS utasítás használatakor ezen a változón keresztül lehet értéket adni a processzor A regiszterének, a gépi kódú rutin lefutása után innen lehet kiolvasni az A regiszter visszatérés előtti utolsó tartalmát. Szerepe a BASIC és a SYS utasítással hívott gépi rutin közötti adatcserére korlátozódik. BASIC-ből a hozzáférés a POKE utasítással, illetve a PEEK függvénnyel lehetséges. (C64, VC20: $030C)

SXREG $07F3 (2035)
Szerepe azonos az SAREG változóéval, de az A regiszter helyett az X regisztert kezeli. Ezt a változót a beépített monitor saját céljaira is használja. (C64, VC20: $030D)

SYREG $07F4 (2036)
Szerepe azonos az SXREG-ével, de az X regiszter helyett az Y regiszterhez tartozik. (C64, VC20: $030E)

SPREG $07F5 (2037)
Szerepe azonos az SAREG-ével, de az állapotregiszter tartalmát lehet vele kezelni. A POKE használatánál óvatosnak kell lenni, mert például az I vagy D bit 1-re átállításának kellemetlen következményei lehetnek (C64, VC20: $030F)

Eljárások

CHRGET $0473 (1139)
A BASIC szövegmutatót (TXTPTR) eggyel megnöveli, majd az általa mutatott karaktert az A regiszterbe tölti. Ha szóközt talált ($20), akkor a fentieket megismétli. Végül beállítja a feltételbiteket az alábbiak szerint:
Z=Ø, ha BASIC utasítás-vég-karaktert (kettőspontot vagy bináris Ø-t) talált, különben Z=1.
C=Ø, ha ASCII számjegyet talált, különben C=1. (C64, VC20: $0073)

CHRGOT $0479 (1145)
A CHRGET belső rutinja. Működésében annyiban különbözik, hogy a szövegmutató értékét nem változtatja. Szóköz esetén a CHRGET címére ugrik. (C64, VC20: $0079)

INDTXT $04A5 (1189)
A TXTPTR által mutatott címről annak tartalmát az A regiszterbe tölti. A feltételbiteket az LDA gépi utasításnak megfelelően állítja be. (C64,

VC20: nincs közvetlen megfelelője)

SYNCHR $9493 (38035)

A BASIC szövegmutató (TXTPTR) által mutatott címen levő karaktert összehasonlítja az A regiszter tartalmával. Ha megegyeznek, a CHRGET rutin végrehajtása után a futás a hívás helyétől folytatódik. Eltérés esetén a SYNTAX ERROR kódjával a hibakezelő eljárásnak adódik át a vezérlés. Az eljárás hívása előtt a várt karakter kódját az A regiszterbe kell tenni. (C64: $AEFF, VC20: $CEFF)

CHKCLS $948B (38027)

A SYNCHR speciális esete. A csukó zárójel jelenlétét vizsgálja a TXTPTR által mutatott helyen. Használata annyiban különbözik a SYNCHR-étől, hogy hívás előtt a zárójel kódját nem kell az A regiszterbe tenni. (C64: $AEF7, VC20: $CEF)

CHKOPN $948E (38030)

Működése, használata megegyezik a CHKCLS-ével, de csukó zárójel helyett nyitó zárójelre vizsgál. (C64: $AEFA, VC20: $CEFA)

CHKCOM $9491 (38033)

Működése, használata megegyezik a CHKCLS-ével, de zárójel helyett vesszőt jelenlétét vizsgálja. (C64: $AEFD, VC20: $CEFD)

GETBYT $9D81 (40321)

A BASIC programból, a TXTPTR által mutatott helytől kezdve, beolvas egy aritmetikai kifejezést és annak aktuális értékét 1 bájtos egész számmá konvertálva az X regiszterbe teszi. A TXTPTR a végrehajtás után az aritmetikai kifejezés utáni első, nem szóközt tartalmazó bájtra mutat. Ha az aritmetikai kifejezés értéke negatív vagy 255-nél nagyobb, a vezérlés ILLEGAL QUANTITY jelzéssel a hibakezelő rutinra adódik. (C64: $B79B, VC20: $D79B)

BARNA LÁSZLÓ

Mi van a bolgár iskolákban?

Bulgáriai látogatásom során alkalmam volt bepillantani az iskolai számítógépes oktatásba. A tanulók 6 és 16 éves koruk között vesznek részt az általános képzésben. Az iskolák feladata az informatikai műveltség megalapozása. Fontos szerep jut tehát a számítógépnek; kormányprogram foglalkozik iskolai alkalmazásával. A számítástechnikai ismeretek megtanítására külön tantárgyat vezettek be, informatika néven. Jelenleg 5400 iskolaszámítógépük van, 1986 végére ez a szám 18 ezerre növekszik, 1995-re pedig a tervek szerint a jelenleginek tízszerese lesz, tehát több mint 54 000!

Az iskolaszámítógépek Pravec típusúak. Talán ismeretes, hogy Bulgária évekkel ezelőtt megvásárolta az amerikai Apple II számítógép licencét, és ezt a gépet gyártja a Pravec nevű kisvárosban épített elektronikai gyárban. Ez a számítógép teljesen kompatibilis az Apple IIe géppel.

Más gépet miniszteri rendelet szerint tilos bevinni az iskolába, így az iskolák gépparkja teljesen egységes. A gépekhez lemezmeghajtó tartozik, amelynek segítségével a mágneslemezen tárolt program másodperceken belül betölthető, a 8 éves gyerekek is nagy biztonsággal kezelik. Nem ismerik tehát a magnetofonos adattárolás nehézségeit és betöltési problémáit. Képernyőként egységesen fekete-zöld monitort használnak (színes monitort csak elvétve láttam), ennek finom felbontóképessége és nem vibráló képe igazán élvezetessé teszi a géppel való foglalkozást.

Az oktatási szoftverek megírására Bulgáriában nincs pályázati rendszer. Általában egyetemi, főiskolai kutatócsoportok vagy egyetemi hallgatók írnak oktatási programokat. Ezekről az oktatási minisztérium katalógust tett közzé, melynek alapján az iskolák megrendelhetik és megvásárolhatják a programokat. Egy lemezen egy teljes programcsomag található. Ára 20 leva körül van, de a tervek szerint a közeljövőben erősen csökkenni fog.

Természetesen a központilag forgalmazott szoftverek mellett jelentős az iskolán belüli programfejlesztés is. A tanárok és diákok maguk is írnak kisebb programokat, melyeket egymás között cserélnek.

Két szófiai és két vidéki általános iskolát látogattam meg. Ezekben 20-30 számítógép állt a tanulók rendelkezésére. Az egyik iskolában a gépek hálózatba voltak kiépítve, de ezt a rendszert — amerikai mintára — oktatógépként használták. A bemutatott programok nagy része a rossz értelemben vett programozott oktatást valósította meg. A tanulók kreatív foglalkoztatását csupán egy helyen láthattam, itt a LOGO nyelvvel ismerkedtek a gyerekek.

A bemutatott didaktikai játékprogramok között láttam néhány kedves ötletet is. Például 8-9 éves gyerekek olyan anyanyelvi játékprogramot használtak, melyben a képernyő szélén feltüntetett szótagokból értelmes bolgár szavakat kellett alkotni. A gép előtt ülő két „versenyző" kislány vitorlás hajón utazott a cél felé. Ha a vitorlán megjelenő vélletlenszerűen választott szótagot jól egészítették ki, a hajó tovább úszott a hullámzó tengeren. A gyerekek nagy kedvvel játszottak a géppel, miközben nyelvtantudásukat is gyakorolhatták.

Az említett iskolákon kívül ellátogattam még a blagojevgrádi Pedagógiai Műszaki Főiskolára is, ahol a tanárképzésben 50 bolgár iskolaszámítógépet és két IBM PC kompatibilis gépet használnak, teljes perifériarendszerrel. Blagojevgrád régi városközpontjának újjáépített házaiban megtekinthettem egy szabadidőközpontot is, melyben 10 iskolaszámítógép is helyet kapott, és az érdeklődő lakosság rendelkezésére áll.

KŐRÖSNÉ
MIKIS MÁRTA

Kedves leendő Szerzőink!

Kérjük, hogy cikkeiket az alábbi kívánalmaknak megfelelően küldjék be:

— 2 példányban gépelve

— a programlisták jól olvasható (nem halvány, elmosódott), normál szélességű karakterekkel, legfeljebb 40 kar/sor formátumban legyenek kinyomtatva,

— az esetleges ábrák feliratai egyértelműek legyenek. A szerzőről az alábbi adatokat kérjük:

— név,
— személyi szám,
— lakáscím,
— munkahelyének neve, címe.

A szerkesztőség fenntartja magának a jogot, hogy a cikkeket — a lényeget nem érintő módon — rövidítse, azokat szabadon felhasználhassa. A szerkesztőség

A DIGITÁL Számítástechnikai Szaküzlet újdonságai Spectrum számítógépekhez

EPROMÉGETŐ ÉS ELLENŐRZŐ (16 k—32 k-hoz) Ára: 4900,— Ft

MEMÓRIABŐVÍTÉS 16 k—48 k Ára: 2800,— Ft

LAPOZÓS MEMÓRIA 48 k—80 k Ára: 250,— Ft (csak a lapozó logika)

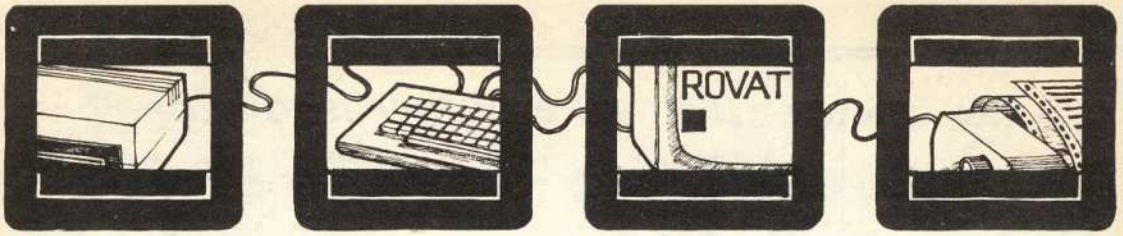
SPECTRUM—TON hangosító Ára: 700,— Ft

KARDOS JÓZSEF műszaki kereskedő
VÉTEL — ELADÁS

Budapest II., Szilágyi Erzsébet fasor 35. 1026
Telefon: 156-231
Nyitva tartás: 9—18 óráig. Szombaton: ZÁRVA

Nemzetközi Számítástechnika — Internacia Komputado

Az 50 országban olvasott magazin **példányonként 30 forintért kapható az SKV Könyvesboltban. Bp. II., Keleti Károly u. 10. Telefon: 158-018**



A C64 GRAFIKUS LEHETŐSÉGEI

A C64 grafikus lehetőségeinek megismeréséhez érdemes tisztáznunk néhány, a normál üzemmóddal összefüggő dolgot.

Ahhoz, hogy a tévé képernyőjén, annak bármely karakterhelyén megjelenjen egy alfanumerikus vagy grafikus jel képe, programozási szempontból a gépnek tudnia kell, hogy hol kezdődik a képernyő-memória, hol kezdődik a színmemória, hol kezdődik a karakter ROM vagy RAM és hogy hol áll a kurzor. Csak ezek ismeretében foglalkozhat a különböző megjelenítési módokból adódó eltérésekkel. A felsoroltak kezdő értékei üzembe helyezéskor, az indítási folyamat alatt automatikusan beállítódnak. Értékeik rendre: 1024, 55296, 53248 (ROM), PEEK (214) a kurzor aktuális sorának, PEEK (221) pedig azon belüli oszlopának a helye.

A keret színét a későbbiekben is figyelmen kívül hagyva, a normál üzemmód grafikai jellemzői a következők: a háttér-papír (kikapcsolt pontok) színe az egész képernyőre választható, de egységes. Értékét POKE 53281, színkód állítja be. A megjelenített karaktertinta (bekapcsolt pontok) színe karakterhelyenként változhat. Értékét a PRINT utasításban leütött színvezérlő karakter vagy a színmemória aktuális helyére bevitt színkód határozza meg. A kép tarkítható az inverz be- és kikapcsolásával, valamint saját karakterkészlet használatával. Az utóbbi ismertetését lásd alább!

Normál üzemmódban a képernyő adott sorának adott oszlopában történő tetszőleges karakter megjelenítéséhez a szokásos PRINT utasításon kívül még két lehetőség:

POKE 1024+40*sor+oszlop, karakterkód: POKE 55296+40*sor+oszlop, színkód, illetve

POKE 214,sor:POKE211, oszlop:SYS58732:PRINT "karakter" ahol sor a 0—24, oszlop a 0—39, színkód a 0—15, karakterkód pedig a 0—255 intevallum egész számainak valamelyike.

A címben szereplő üzemmódok, illetve a saját karakterkészlet használatához különböző átkapcsolások, beállítások szükségesek. A B/K feladatok ellátásáért a VIC chip a felelős. Információit mindig a memória valamely 16 k-s szeletéből veszi a színmemória kivételével, mely nem mozgatható, mindig az 53296 címen kezdődik, és 1000 bájt hosszú. Azt, hogy a VIC melyik 16 k-s szeletből veszi az információt, a CIA—2 A kapujának 0. és 1. bitjei határozzák meg. Az A kapu címe 56576. Ahhoz, hogy a szeletet megváltoztathassuk, előbb az A kapu adatirány-regiszterének alsó két bitjét kell magasra állítanunk. Az A kapu adatirány-regiszterének címe 56578. A szelet átállítása a

POKE 56578,PEEK(56578) OR3 és a

POKE 56576,(PEEK(56576) AND252)ORSZ

parancsokkal megoldható, ahol SZ a 0,1,2,3 értékeket veheti fel. Ekkor a VIC által használt szelet rendre a 49152, 32768, 16384, 0 címeken kezdődik; az alaphelyzet az utolsó!

A képernyő-memória adott szeleten belüli kezdetét az 53272 cím felső négy bitje határozza meg. Amennyi ennek — mint négybites egész számnak — az értéke, annyi k-t adjunk a szelet kezdetéhez ahhoz, hogy a képernyő-memória kezdő címét megkapjuk. Alaphelyzetben ez az érték 1, így a képernyő-memória a 1024 címen kezdődik.

Az átállításnál vigyázzunk, mert az 53272-es cím alsó négy bitje az aktuális karaktermemória 2 k-s blokkjának szeleten belüli kezdetét határozza meg, pontosabban a 0. bitet figyelmen kívül hagyjuk. Így tehát a képernyő-memória kezdetét szeleten belül 1 k-s lépésekben állíthatjuk a

```
0 GOTO10
1 ************************************
2 *        TOBBSZIN UZEMMOD!         *
3 ************************************
10 POKE56,55:POKE55,255:CLR
15 CI=14336:V=53248
20 POKE646,9:PRINT"J"CHR$(8)
25 FORC=256TO263:POKECI+C,0:NEXT
30 FORC=CITOCI+7:POKEC,27:NEXT
35 FORC=8TO15:READA%:POKECI+C,A%:NEXT
40 DATA 24, 60,102,126,102,102,102,  0
45 POKEV+32,5:POKEV+33,15
50 POKEV+34,12:POKEV+35,11
55 POKEV+22,216:POKEV+24,31
60 FORC=0TO1000:POKE1024+C,0:NEXT
65 POKE646,0
70 PRINTTAB(9)"                    "
75 PRINTTAB(9)" AAAAAAAAAAAAAAAAAA "
80 PRINTTAB(9)"                    "
85 WAIT203,62:POKE1024,0:SYS1024
```

1. program

```
0 goto10
1 ************************************
2 *   bovitett hatterszin uzemmod!   *
3 ************************************
10 v=53248:Print"■"chr$(8);
20 Pokev+32,12:Pokev+33,12:Pokev+24,23
30 Pokev+34,2:Pokev+35,6:Pokev+36,0
40 Pokev+17,Peek(v+17)or64
50 Print"az elet szep!"
60 Print"AZ ELET SZEP"chr$(161)
70 Print"■az elet szep!"
80 Print"■AZ ELET SZEP"chr$(161)
90 gosub140
100 ift)<t+6then100
110 ifa%=7then9ocub140
120 reada:Pokev+36,a:a%=a%+1
130 t=ti:goto100
140 a%=0 restore
150 forl=0to200 nextl:return
160 data11,12,15,15,12,11,0
```

2. program

```
20 GOSUB330:COLOUR0,0
25 REM ********************************
26 REM *    NYOMTATO KIRAJZOLASA!     *
27 REM ********************************
30 FILL17,2,35,8,160,7:FCOL17,3,30,7,9
50 FCHR17,3,5,7,0:FILL18,37,3,4,0,9
70 FILL20,37,1,5,160,7
80 FCHR17,8,29,1,228
90 PRINTAT(2,17)"■■CHR$(164)
100 PRINTAT(8,23)"■■COMMODORE";
105 PRINT"            MPS801"
110 PRINTAT(9,18)"■U L":GOSUB430
130 PRINTAT(8,16)"■■■ "DUP(" ",21)" ■"
131 :
132 REM ********************************
133 REM *          NYOMTATAS!          *
134 REM ********************************
140 FORC=0TO15:CX=15-C
141 FORL=0TO20:NEXTL
150 MOVE16,8,25,1,CX,8:NEXTC
151 :
160 READX$:X=LEN(X$):IFX$="*"THEN240
170 X$=LEFT$(X$+DUP(" ",21),21)
180 FORC=0TOX
185 IFMID$(X$,C+1,1)<>" "THENPLAY2
186 FORL=0TO20:NEXTL
190 RIGHTW18,9,23,1:NEXTC
200 FORC=0TOX:LEFTW18,9,23,1:NEXTC
210 UPW0,10,21,17:PRINTAT(10,16)"■■"X$
230 GOTO160
231 :
240 PRINT"■":END
245 :
270 DATA "    SIMON'S BASIC"
280 DATA "  KARAKTER GRAFIKAI"
290 DATA "    DEMO  PROGRAM"
300 DATA "","  (C) BY FOLDI ENDRE"
320 DATA "",*
321 :
322 REM ********************************
323 REM *        BEALLITASOK.          *
324 REM ********************************
329 REM **** "@" KARAKTER ATIRASA. ***
330 MEM:DESIGN2,$E000
340 @BBBBBBBB
350 @........
360 @BBBBBBBB
370 @........
380 @BBBBBBBB
390 @........
400 @BBBBBBBB
410 @........
420 RETURN
425 REM *** HANG GENERATOR BEALL. ***
430 VOL15:WAVE1,10000000
460 ENVELOPE1,1,1,1,0:MUSIC1,"■IC5■■3"
470 RETURN
```

3. program

POKE 53272,(PEEK(53272) AND15) OR KE

paranccsal, ahol KE a 0, 1, 2, . . . ,15 egész számok valamelyike.

A karaktermemória 2 k-t foglal el. Ennek szeleten belüli kezdetét a

POKE 53272,(PEEK(53272) AND240) OR KA

paranccsal állíthatjuk 2 k-s lépésekben (a 0. bit értéke ezért közömbös), ahol KA a 0, 2, 4, . . . ,14 egész számok valamelyike lehet.

Alapállapotban a karakter ROM a B/K regiszterek mögött, alatt helyezkedik el az 53248—56832 címeken. Ezt a területet csak a VIC érheti el. Saját karakterek használatához a karakter ROM-ot a RAM valamely, az előzőekben ismertetett 2 k-s részére másoljuk át, majd a szükséges karakterképek megváltoztatása után hajtsuk végre az átkapcsolást. A másolás alatt megszakításkérés nem adható ki, a B/K regisztereket ki kell kapcsolni, ezért a billentyűzet is tiltva van! A másolás előtt tehát adjuk ki a

POKE 56334,PEEK(56334) AND254

— megszakítás tiltva

POKE 1,PEEK(1)AND251 — B/K ki parancsokat,

majd a másolás végén a

POKE 1,PEEK(1)OR4 — B/K be,

POKE 56334,PEEK(56334) OR1 — megszakítás engedélyezve

parancsokat. Ezek után következhet a karakterkészlet új alakjának betöltése az adott helyre, majd az átkapcsolás az új karakterkészletre. Az új karakterkészlet helye általában a BASIC terület elején van, ezért először a már ismert módon ennek beállításáról gondoskodjunk.

Többszínű karakter-üzemmód

Az 53281-es cím tartalma a papír vagy a háttér-0 színt határozza meg, az egész képernyőre érvényesen. Az 53282 és 53283 címek a szakirodalom szerint a háttér—1 és háttér—2 színeket adják, mi azonban inkább tinta—1 és tinta—2 színeknek neveznénk őket, hiszen miattuk lesz többszínű a karakter! A tényleges tinta színe karakterhelyenként változhat, értékét a színmemória aktuális helyének alsó három bitje határozza meg a 0—7 színkódokkal!

Mit jelent ez a sok szín? A karakter nyolc bájtjának bitjeit csoportosítsuk kettesével balról jobbra! A kapott bitpárok értéke 00, 01, 10, 11 lehet. A helyükön a képernyőn lévő két képpont színe rendre háttér—0, tinta—1, tinta—2 és tintaszínű lesz, ha a többszínű üzemmódot bekapcsoltuk, és a színmemória aktuális helyének 4. bitje is magas. A kép olyan, mintha a karakter „közepét" jelentő 11 bitpár tintaszínét karakterenként változtathatnánk, a karakterkép „széleit" jelentő 01, illetve 10 bitpárnál pedig a 00 által jelölt háttérhez viszonyítva két különböző színnel kihúznánk a karakter bal és jobb oldalát.

A többszínű üzemmód használatához az 53270-es cím 4-es bitjét kell magasra állítani, azaz a

POKE 53270,PEEK(53270) OR16

paranccsal be-, a

POKE 53270,PEEK(53270) AND239 paranccsal kikapcsolhatjuk.

Ha a bekapcsolás után az aktuális színmemória színkódja a 0—7 tartományba esik, akkor a karakter normál üzemmódban jelenik meg a választott színben, ellenkező esetben többszínű üzemmódban, de a tinta színét csak az alsó három bit (0—7) határozza meg. Így a többszínű és normál megjelenítés a színmemória aktuális helye 4. bitjének magasra-alacsonyra állításával karakterenként vezérelhető, azaz a két megjelenítési mód egy képernyőn belül is keverhető. Természetesen ebben az üzemmódban is használhatunk saját tervezésű karakterkészletet.

Bővített háttérszín-üzemmód

A többszínű üzemmódban az egész képernyőre érvényes háttér—0 szín, illetve tinta—1 és tinta—2 szín mellett csak a tinta színét változtathatjuk karakterenként, most azonban négy háttérszínt definiálhatunk. Azt, hogy az aktuális helyen a karakter milyen színű háttér előtt jelenjen meg, a négy szín közül karakterhelyenként megválaszthatjuk, a tinta színével együtt. A tinta színét a PRINT utasításba vitt színvezérlő karakterekkel vagy a színmemória aktuális pozíciójára beírt színkódokkal határozhatjuk meg.

A karakterenként választható négy háttérszín kódját (háttér—0 — háttér—3) az 53281, 53282, 53283, 53284 címekre írhatjuk be. Az üzemmód beállításához az 53265 cím 6. bitjét kell magasra állítani, azaz a be-, illetve kikapcsolást a

POKE 53265,PEEK(53265) OR64 illetve a

POKE 53265,PEEK(53265) AND191 parancsok végzik el.

A sok színnek ára most is van, ugyanis megjeleníteni csak a 0—63 képernyőkódú eredeti vagy saját karaktereket tudjuk, mert a karakterkód 7. és 6. bitjének értéke együttesen határozza meg az aktuális karakterhelyen érvényes háttér színét. Ha két bit együttes értéke 00, 01, 10, 11, akkor a ka-

rakter az aktuális tintaszínnel rendre a háttér—0, . . ., háttér—3 színű papír előtt jelenik meg.

A programok

Az *1. program* azt mutatja be, hogy miként lehet többszínű üzemmódba kapcsolt képernyőn normál karaktert vagy szöveget megjeleníteni. Magyarázata:

- 10 a BASIC terület végének beállítása
- 15 szelet és VIC báziscímének beállítása
- 20 tintaszín-beállítás, karakterkészlet-váltás
- 25 szóközkarakter nullázása
- 30 @ átírása
- 35 nagy A helyén nagy A készítés (több karakter nincs!)
- 45 képernyőszínek beállítása
- 55 többszínű üzemmód és szeletátkapcsolás
- 60 képernyőtöltés
- 65 fekete tinta
- 70 kiírás
- 80 billentyűre vár, alaphelyzetbe vissza

A *2. program* a bővített háttérszín-üzemmódot illusztrálja. Magyarázata:

- 10 VIC báziscím, karakterkészlet-váltás letiltása
- 20 képernyőszín, kisbetű
- 30 háttérszínek
- 40 bővített háttérszín-üzemmód bekapcsolása
- 50 normál kiírás
- 60 shiftelt kiírás
- 70 inverz kiírás
- 80 shiftelt inverz kiírás
- 90 alaphelyzetbe állít, vár
- 100 villogtat
- 160 színkódok

A *3. program* karaktergrafikai összefoglaló, mely egyben a legelterjedtebb BASIC-bővítés, a SIMON'S BASIC lehetőségeinek egy részét is felvillantja. Mozgó grafika, egy MPS801-es nyomtatót szimulál. Begépelése előtt töltsük be és futtassuk a SIMON'S BASIC programot. Magyarázata:

- 20 képernyőszín-beállítás
- 30— 90 nyomtatótest kirajzolása
- 100—105 feliratozás
- 110 nyomtatófej-rajzolás, hanggenerátor hívása
- 130 papírcsíkrajzolás
- 140—150 üres papír megjelenítése
- 160—230 szöveg nyomtatása
- 270—320 a nyomtatandó szöveg
- 330—420 a nyomtatótest csíkos részeit előállító karakter tervezése
- 430— a hanggenerátor beállítása, a 460-as sor inverz bal alsó negyedköre F2 lenyomására jelenik meg.

FÖLDI ENDRE — ÉNEKES FERENC

DIÁK magazin ROVAT

Kódkonverter

Az alábbi kis assembly program segítségével bármely, 0—65535-ig terjedő decimális számot átválthatunk hexadecimális számrendszerbe. A programot töltsük be a gépi kódú területre, vagy a címeket átírva és a BASIC területet eltolva, a BASIC elejére. A BASIC program az A változó értékét átadja a gépi kódú résznek. A-t itt binárisan ábrázoljuk, ezért kell &7904-re &B2-t tölteni. A gépi rész A-t átváltja hexadecimálissá, majd A$-be tölti.

```
10:"M":WAIT :CLS :INPUT A:B=INT (A/
   256):C=A-B*256:POKE &7904,&B2,B,
   C:CALL &7E01,A:PRINT A:END
7E01    68 LDI UH  78
7E03    6A LDI UL  C8
7E05    84 LDA XH
7E06    BE SJP     7E10
7E09    61 SIN U
7E0A    84 LDA XH
7E0D    BE SJP     7E1E
7E0E    61 SIN U
7E0F    84 LDA XL
7E10    BE SJP     7E10
7E13    61 SIN U
7E14    84 LDA XL
7E15    BE SJP     7E1E
7E18    61 SIN U
7E19    85 LDI A   00
7E1B    2E STA (U)
7E1C    9A RTN
7E1D    F1 AEX
7E1E    89 ANI A   0F
7E20    87 CPI A   0A
7E22    83 BCS +   03
7E24    83 ADI A   30
7E26    9A RTN
7E27    83 ADI A   36
7E29    9A RTN
```

A programot a saját készítésű assembly-disassembly programmal csináltam, és kiírattam KA 150-es nyomtatón. A programot DEF H-val futtathatjuk, majd írjuk be az átváltandó számot.

BALÁZS GYÖRGY

Monológ

Végre ezt is megértük! Hogy mit? Hát te nem vettél részt a számítástechnikai OKTV iskolai fordulóján? De igen? Akkor biztosan egyetértesz velem, hogy ez volt az utóbbi évek legszínvonalasabb versenye. Már úgy értve, hogy a feladatlap . . . Azt mondod, neked is tetszett? Hogy nem ragaszkodott kizárólag a HT ismeretéhez? Hát persze, talán ez volt a legnagyobb előrelépés a régiekhez képest. Mi? Hogy már nemcsak azt kellett kitalálni, minek bonyolították feleslegesen a feladatot holmi változótípusokkal!

De nem volt rossz az általános ismeretek tesztje sem. Sőt! Belekóstoltattak még egy másik nyelvbe is, ez volt az egyik legérdekesebb feladat. Szinte felüdülés volt a sok BASIC után. Hogy nem is volt benne olyan sok? Ja, igen. Mit szólsz az algoritmusos feladathoz? Kissé bizarr volt, nem? De azért jó ötlet volt ez is. Annak örültél, hogy jól áttekinthető programokat kellett elemezni? Na igen, ez régebben nem volt túlságosan jellemző.

Reménykedjünk, hogy ezentúl legalább ilyen színvonalúak lesznek a számítástechnikai versenyek. Most meg hová rohansz? Mi? Bejutottál a második fordulóba? Gratulálok! — mondta X, és új partner után nézett, akinek tovább ecsetelheti a feladatlap nagyszerűségét.

PALLAGI LÁSZLÓ
Vörösmarty Gimnázium
Érd

SEGÉDPROGRAM KEZDŐK RÉSZÉRE

A program segítségével meghatározhatjuk a HT-gépeken a grafikus karakterek kódját. A grafikus karakter hat részből áll:

1	2
4	8
16	32

Kódját megkaphatjuk, ha 128-hoz hozzáadjuk azokat a számokat, amelyek a világító mezőkben találhatók. Ennek kiszámítását könnyíti meg a program. Futtatásakor a kiírt számok egy-egy mezőt jelölnek:

1	2
3	4
5	6

A kivilágítani kívánt pontoknak megfelelő számokat üssük le! Ha befejeztük, akkor a NEW LINE billentyű lenyomása után megkapjuk az elkészült karakter kódját és a karaktert is, eredeti nagyságban.

A programot akkor érdemes használni, ha több grafikus karakter felhasználásával akarunk képernyőt tervezni. Ilyenkor célszerű a készülő programunk után másolni (MERGE — lásd a µM 1985/1. szám 8. oldalán); használata után pedig DELETE 60000—60130 utasítással törölhetjük.

JOÓ ZOLTÁN
Piarista Gimnázium
Budapest

DIÁK HT-1080z ROVAT

```
60000 REM GRAFIKUS
  KARAKTEREK HT-N
60010 DATA 1,64,2,65,4,
 128,8,129,16,192,32,193
60020 FOR A=1 TO 6:
 READ A(A),B(A):NEXT A
60030 CLS:PRINT:PRINT"12":
 PRINT"34":PRINT"56"
60040 PRINT"VEGE:NEW LINE"
60050 B=128
60060 A$=INKEY$:
 IF A$="",GOTO 60060
60070 IF A$=CHR$(13),
 GOTO 60120
60080 X=VAL(A$)
60090 IF C(X)=0,
 B=B+A(X):C(X)=1
60100 IF B(X)>0,
 PRINT@B(X),CHR$(191);
60110 GOTO 60060
60120 PRINT@650,B,CHR$(B)
60130 IF INKEY$="",
 GOTO 60130
60140 RUN 60000
```

Assemblerek, cross-assemblerek

6. Az operátorkészlet I.

A következőkben megadjuk a rendszer leírásához használt operátorok definícióit. Az operátorokkal kapcsolatban elöljáróban két dolgot kell kiemelni, illetve ismételni.

Egyrészt a bemutatott megoldás csak egy a sokféle lehetséges közül. Több olyan operátor fog szerepelni (például a 13., 16., 19. operátorok), amelynek a feladata, működése annyira összetett, hogy talán célszerűnek látszana további, elemibb operátorok együttesére felbontani őket. Mégsem tartottuk indokoltnak az operátorok ilyen mértékű felaprózását; célunk inkább a rendszer működési folyamatában a nagyobb léptékű összefüggések megvilágítása. Így alakult ki a bemutatásra kerülő heterogén operátorhalmaz.

Másrészt, az előző fejezetek tárgyalásmódjának az általánosíthatóságot hangsúlyozó jellegével szemben, egyes operátorok működésének leírásakor már kevésbé tudunk eltekinteni a kiválasztott nyelv, az Intel 8080 mikroprocesszor assembly nyelvének sajátosságaitól. Ezeknek a hatása a korábban tett feltételezések és korlátozások mellett az operátorok felvételére helyenként döntő jelentőségű. Természetesen más mikroprocesszor más nyelvéhez írt fordítóprogram hasonló feladatokat ellátó operátorokat igényel, ha ezek minden részletben nem is egyeznek meg az itt bemutatottakkal.

Ebben és a következő részben összesen 24 operátort definiálunk, amelyek segítségével építjük fel a rendszer kapcsolatábráját.

Nyolcrészes sorozatunk a mikroszámítógépek assemblereiről, cross-assemblereiről szól. Célja, hogy a cross-assemblerek példáján keresztül megismertesse az olvasót az assembler programok működésével. A bemutatáshoz a rendszermodellezési eszközöket használjuk fel, s így készítjük el az assemblerek működésének egy szabványosított algoritmusát, modelljét. E cikkünk a sorozat hatodik része.

1. (Cross-)assemblerek és a rendszermodellezés
2. Az Intel 8080 assembly nyelv
3. A (cross-)assembler program mint rendszer
4. A rendszer működése I. — A fordítás két menete
5. A rendszer működése II. — Táblák és adatterületek
6. Az operátorkészlet I.
7. Az operátorkészlet II. — A rendszer kapcsolatábrája
8. Példák a rendszer működésére

1. operátor: paraméterlekérdező és kezdeti-érték-beállító *(1. ábra)*

Feladata:

— a programindítás után lekérdezi a rendszer bemeneti paramétereit:

- a bemeneti periféria típusát, illetve mágneslemez esetén a forrásprogramfájl nevét,
- a kimeneti periféria típusát, illetve mágneslemez esetén a tárgyprogramfájl nevét,

— beállítja a rendszerváltozók és egyéb változók kezdeti értékét, azaz:

- a PASS, DEF, MIND, IF1, IF2 az A1 állapotnak megfelelő értékeket veszik fel,
- a pufferterületek és táblák mutatói a megfelelő kezdeti értékre állítódnak be,
- a sorfeldolgozási ciklussal kapcsolatos indikátorok is felveszik a kezdeti értéküket,

— kijelöl a mágneslemezen előre meghatározott nagyságú munkaterületeket a forrásprogram, a modul- és a makrócímketáblák számára,

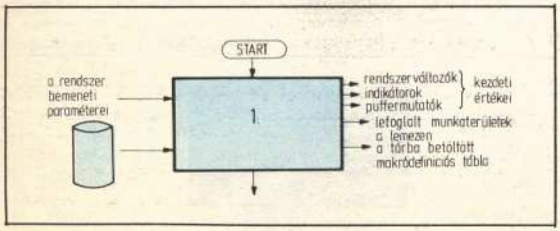
— betölti a makródefiníciós táblát (vagy annak az utolsó rekordját) a központi tárba.

2. operátor: forrásprogram-olvasó nem lemezes bemeneti perifériáról *(2. ábra)*

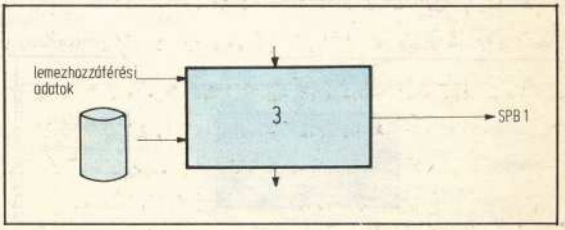
Feladata: egy utasítássort olvas be a bemeneti perifériáról az első bemeneti pufferterületre (SPB1).

3. operátor: forrásprogram-olvasó mágneslemezről *(3. ábra)*

Feladata: egy utasítássort olvas be SPB1-re mágneslemezről, akár mint bemeneti perifériáról, akár mint az első menetben a le-

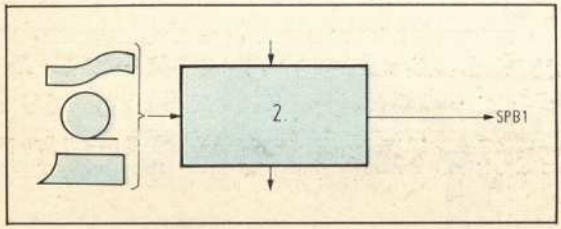


1. ábra

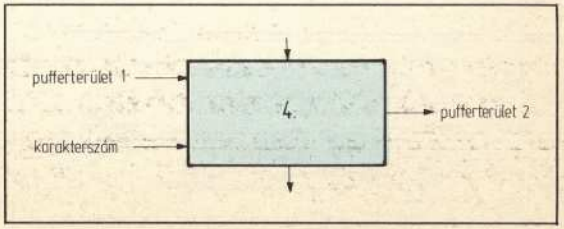


3. ábra

2. ábra



4. ábra



mezre felvitt kifejtett forrásprogram átmeneti tárolójáról, akár mint a makródefiníciók tárolóhelyéről.

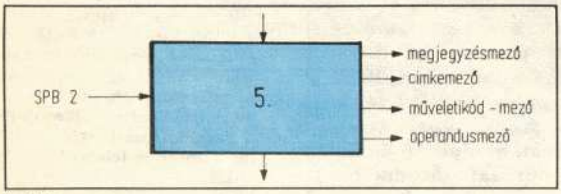
4. operátor: karakterlánc-mozgató *(4. ábra)*
Feladata: egy adott pufferterületről adott számú karaktert visz át egy másik pufferterületre.

5. operátor: egy forrásprogramsort mezőkre bontó *(5. ábra)*
Feladata: az SPB2 pufferterületen tárolt forrásprogramsort logikai alkotóelemeire bontja fel, s kitölti az egyes mezőknek megfelelő önálló pufferterületeket.
A mezők: megjegyzésmező, címkemező, műveletikód-mező, operandusmező. Az operandusmezőbe csak a valódi operandustartalmat viszi át, az esetleg az operandus(ok) után vagy helyett álló megjegyzés helyét szóközkarakter tölti ki. A mezőhatárok előre rögzítettek.

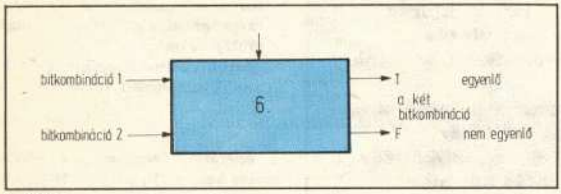
6. operátor: több-bites párhuzamos komparátor *(6. ábra)*
Feladata: a bemenetére adott két bitkombináció egyenlőségét megállapítani. Ez jelentheti két 1 bájt hosszú karakter vagy egy 2 bájtos egész típusú változó értékének és egy konstansnak az összehasonlítását.

7. operátor: formális-aktuális paramétercserét végző *(7. ábra)*
Feladata: makrókifejtés esetén — ha a sor nem a kezdő, a MACRO direktívát tartalmazza — elvégzi a definícióban szereplő formális paramétereknek a hívásnál megadott aktuális paraméterekre való cseréjét. Elvileg mind a címke-, mind a műveletikód-, mind az operandusmezőben megengedett a formális paraméterek használata, fordítóprogramunkban azonban csak az operandusmezőben tesszük lehetővé a paramétercserét. Az operátor egyenként megvizsgálja, hogy az operandusmezőben — ha ez nem üres — szereplő, s egymástól vesszővel elválasztott egyszerű kifejezések nem egyeznek-e meg a megszakítási hely és makróparaméterek veremtárában az aktuális bejegyzésben tárolt valamelyik formális paraméterrel. Ha valahol egyezést talál, akkor a kifejezés helyére a megtalált formális paraméter sorszámával azonos sorszámú aktuális paramétert teszi be, amelyet ugyanebből a verembejegyzésből olvas ki. Látható, hogy rendszerünkben nem engedjük meg formális paramétereknek összetett kifejezésekben való megjelenését, bár az operátor működése kiterjeszthető lehetne erre az esetre is.
Az aktualizált operandusmezőt elhelyezi az SPB2 pufferterület megfelelő helyére is, hogy a lemezre az aktualizált forrássor kerüljön fel.
Ha a MIND rendszerváltozó értéke nulla, akkor az operátor hívása után azonnal be is fejezi a működését.

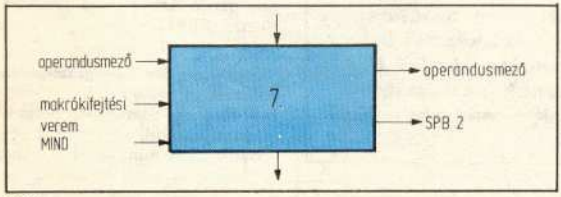
8. operátor: címketáblában kereső és címkeinformációt szolgáltató *(8. ábra)*
Feladata: a modul- vagy az aktuális makrócímketáblában egy megadott szimbolikus név meglétét dönti el, s ha a keresés sikeres, akkor a névhez tartozó címkebejegyzést hozzáférhetővé teszi. Modulcímketáblában való keresés esetén a keresést nemcsak az MLB pufferterületen, hanem a már a lemezre kivitt címkerekordoknak a DFB3 pufferterületre való egyenkénti visszahívásával is végzi. Makrócímkék esetén a kutatási terület a DFB4 puffer (K = 1). A megtalált címkebejegyzést a pufferterületen belüli kezdőcím szolgáltatásával teszi hozzáférhetővé.
A második menetben, ha a keresés sikertelen, akkor beállít egy hibaüzenet-indiká-



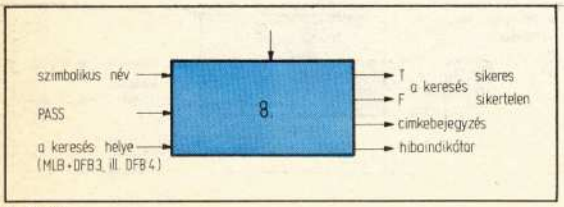
5. ábra



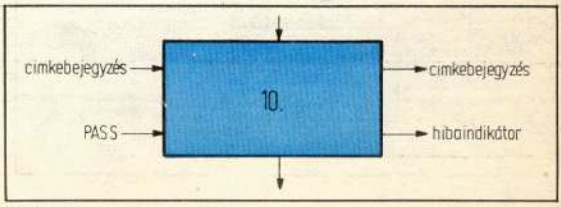
6. ábra



7. ábra

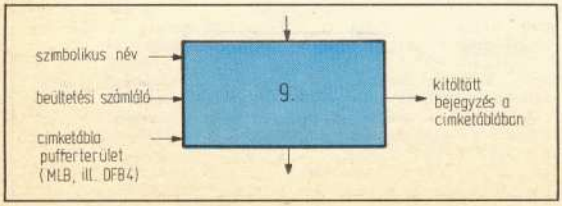


8. ábra

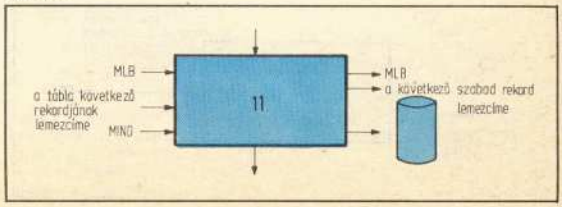


10. ábra

9. ábra



11. ábra



tort, jelezvén, hogy a szimbolikus név az első menetben azért nem került be a táblába, mert a tábla számára fenntartott munkaterület betelt.

9. operátor: címkeinformációt a címketáblába elhelyező *(9. ábra)*

Feladata: a megadott címke vagy azonosítónév alapján bejegyzést készít a modulvagy az aktuális makrócímketáblában. A bejegyzés az MLB, illetve a DFB4 pufferterület szabad területének elejére kerül, az operátor a szimbolikus nevet helyezi el, s relatív címkét feltételezve kitölti a flag- és az értékadatokat. A bejegyzés elkészítése után a puffer szabadterület-mutatóját előreállítja.

Ha akár a makrócímkék, akár a modulcímék esetén a lemezen lefoglalt munkaterület már betelt, akkor a szimbolikus név nem kerül be a táblába (a megengedettnél több modul-, illetve makrócímke szerepel a programban).

10. operátor: többszörös címkedefiniálást kezelő *(10. ábra)*

Feladata: nem megengedett többszörös címkedefiniálás esetén érvényteleníti a címketáblában szereplő bejegyzést. A korábban elmondottak értelmében minden olyan esetben, amikor a fordítóprogram az első menetben a címkemező tartalmával megegyező nevű és nem SET direktivával generált bejegyzést talál a megfelelő címketáblában, hiba van a forrásprogramban. Az operátor a hibát jegyzi be a címketáblába azzal, hogy a flaget a többszörös definiálást jelző értékre (esetünkben 5-re) állítja be, s a címke értékét nullára változtatja meg.

Ha a második menetben hívjuk az operátort, akkor azt vizsgálja meg, hogy a bejegyzésben a címkeflag egyenlő-e a többszörös definiálást jelző értékkel (5-tel). Ha igen, akkor a többszörös definiáláshoz tartozó hibaindikátort a hiba jelzésére állítja be.

11. operátor: a címketáblát lemezre író *(11. ábra)*

Feladata: a modulcímketábla pufferterületének (MLB) beteltségétől függően egy rekordnyi címkeadatot ír ki a lemezre, a címketábla számára fenntartott munkaterület soron következő rekordjaként.

A rekord kiírása után módosítja a munkaterület következő szabad rekordjának kezdőcímét, s az MLB puffermutatóját a pufferterület elejére állítja be, jelezve, hogy a puffer üres, illetve tartalma érvénytelen.

Gyorsíthatjuk a rendszer működését, ha a fenti vizsgálatot az operátor csak MIND=Ø esetén hajtja végre, mert makrókifejtés esetén MLB-re írás nem történik, így az nem is telhet be.

Ha K>1, akkor az operátornak ugyanezeket a feladatokat makrókifejtés esetén a makrócímketáblára is végre kell hajtania.

12. operátor: makróhívást felismerő *(12. ábra)*

Feladata: a műveletikód-mezőben levő karaktersorozatot összehasonlítja a makródefiníciós táblában szereplő makrónevekkel. Ha valahol egyezést talál, akkor a makrónévhez tartozó bejegyzést hozzáférhetővé teszi a definíciós táblában.

13. operátor: makrókifejtést indító *(13. ábra)*

Feladata: a makrókifejtés indításával kapcsolatos, a fordítási menet által előírt műveletek elvégzése. Mivel a makrókifejtés indításának folyamatával a 2. és az 5. részben részletesen foglalkoztunk, itt csak vázlatosan soroljuk fel a szükséges műveleteket.

Az operátor által végzett műveletek az első menetben:

— Ha a megszakított szinten is makrókifejtéses állapotban van a rendszer (MIND≠Ø), akkor a hozzá tartozó makrócímketáblát átmenetileg lezárja és kiírja a LEMEZCÍMi2 (K>1), ill. LEMEZCÍMi1 (K=1) által meghatározott kezdőcímű lemezterületre, DFB4 mutatóját a terület elejére állítja.

— Mélyíti a megszakítási hely és makróparaméterek, valamint a makrócímketáblaveremtárakat: nevezetesen az előzőbe a megszakítási hely és az aktuális paraméterek beírásával egy nem lezárt bejegyzést készít, az utóbbiba pedig az LMCT megnövelt értékét írja be. Érvényes „megszakítási hely" adat csak abban az esetben van, ha a megszakított szinten a bemeneti periféria mágneslemez. (A megszakítási hely meghatározása külön figyelmet kíván, mert a sorfeldolgozási ciklussal párhuzamosan indított olvasás ezt a sort már behozta SPB1-re, így a pillanatnyi lemezadat- és rekordelhelyezési mutatók már nem az elmentendő állapotot tükrözik.)

— Ha a megszakított szint a nulladik és a bemeneti periféria nem mágneslemez, akkor a sorfeldolgozási ciklussal párhuzamos (a következő sort szolgáltató) olvasás befejeződése után elmenti a beolvasott sort SPB1-ről SPB3 területre.

— A makródefiníciós táblából nyert, s a makródefiníciónak a lemezen való elhelyezkedését megadó kezdőcím alapján a 3. operátor belső hívásával beolvassa a definíció első — a MACRO direktívát tartalmazó — sorát az SPB1 területre.

— MIND értékét eggyel növeli.

Az operátor által végzett műveletek a második menetben:

— növeli LMCT értékét és mélyíti a makrócímketábla-vermet,

— az LMCT új értéke által meghatározott lemezcímről beolvassa az új makrókifejtési szinthez tartozó makrócímketáblát, illetve ennek első rekordját,

— MIND értékét eggyel növeli.

14. operátor: műveletikód-kereső és műveletikód-információkat szolgáltató *(14. ábra)*

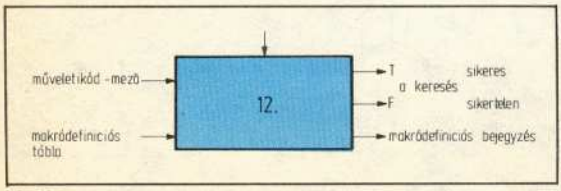
Feladata: a műveletikód-mező tartalmával azonos nevű bejegyzést keres a műveletikód-táblában vagy annak csak egy kijelölt részén. Ha a keresés sikeres, akkor a bejegyzésben szereplő műveletikód-információkat hozzáférhetővé teszi. Ha a keresés sikertelen, akkor beállít egy ennek jelzésére szolgáló hibaindikátort.

A keresés tartománya — tehát, hogy az egész táblában vagy annak csak egy részében kutasson — például a tartomány kezdőcímének és a megvizsgálandó műveletikód-bejegyzések számának megadásával jelölhető ki. (A kapcsolatábrában a keresendő karakterláncokat tüntetjük fel ezen adatok helyett.)

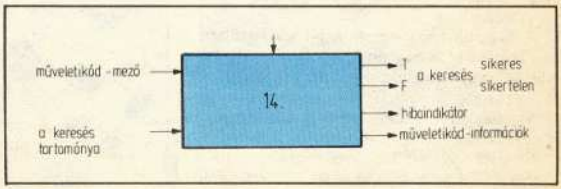
15. operátor: direktívafelismerő *(15. ábra)*

Feladata: a műveletikód-információk alapján eldönti, hogy a műveletikód-mezőben gépi utasítás vagy direktíva mnemonikus kódja szerepel-e.

VÁRGEDŐ TAMÁS

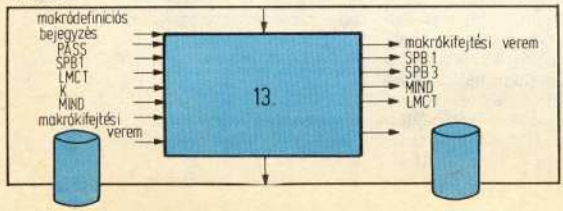


12. ábra

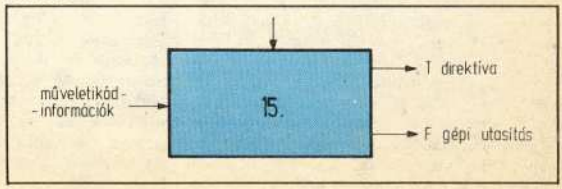


14. ábra

13. ábra



15. ábra



Hogyan működnek a lapos kijelzők?

A képernyős megjelenítők rendkívül fontos szerepet töltenek be a számítógép-használat során, hiszen azok teremtik meg az ember számára a géppel való vizuális kapcsolatot. A napjainkban használatos adatmegjelenítőknek szinte kivétel nélkül katódsugárcsöves képernyőjük van. Ezek hátránya viszonylag nagy energiaigényükön túl a terjedelmes kivitel is.

A kutatók megpróbáltak olyan új utakat, elveket keresni, amelyek lehetővé teszik a képernyős megjelenítők minél laposabb kivitelű előállítását. Az eddig elért eredmények tükrében tekintsük át — a teljesség igénye nélkül — azokat a részben vagy teljesen új működési elveket, amelyeket már sikerrel alkalmaztak a tömeggyártásban is.

Kristályos folyadék (KRIFO) kijelzők

Ma ez a leggyakrabban használt kijelzési technika a lapos kijelzőkben. Nem számít lényegesen újnak, hiszen a zsebszámológépek és a kvarcórák kijelzőjeként már találkozhattunk velük. Újdonságot csupán a kijelzőfelület megnövekedése jelent. E technikában a japán ipar áll az élen: színes, „hordozható" tévét is gyártottak ilyen típusú képernyővel.

A KRIFO-képernyők a passzív kijelzők családjába tartoznak, mivel — ellentétben a más elveken működő kijelzőkkel — nincs saját fényük, hanem a környezeti fényt hasznosítják. A működési elv azon alapul, hogy a KRIFO megváltoztatja optikai tulajdonságait a ráadott feszültség hatására. Ilyen típusú kijelző felépítését láthatjuk az *1. ábrán.*

Kövessük végig a fény útját a kijelzőben. A fény először egy polarizációs szűrőn halad át, majd az elektródamátrixszal ellátott KRIFO-rétegbe jut. További útja során egy második polarizációs szűrőn áthatolva, a végállomást jelentő tükörről visszaverődve, a fentebb leírt utakat újból végigjárva jut a szemünkbe. Az ily módon visszaverődő fény tehát akadálytalanul éri el szemünket. Ha azonban az elektródamátrix egyik pontjára feszültséget adunk, akkor megváltozik a fény polarizációs iránya, így annak áthaladása lehetetlenné válik, és egy sötét pont keletkezik a mátrixpontban. Az elektródamátrixot sűrű dróthálóként képzeljük el. Belátható, hogy az így nyert kép annál finomabb felbontású, minél sűrűbb a háló.

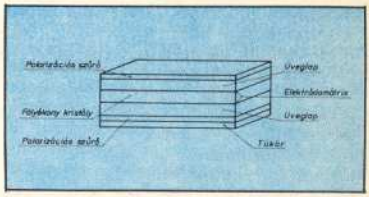
Az ilyen elvű kijelzők rendkívül laposak, viszonylag kis súlyúak. Alacsony az energiaigényük. E szempontok miatt különösen alkalmasak „hordozható", táskakivitelű számítógépekhez. Jóllehet kevés környezeti fényt igényelnek, nagy megvilágítású környezetben is használhatók. Áruk viszonylag alacsony. Hátrányuk, hogy a többi kijelzőkhöz képest kontrasztszegényebb képet adnak, és viszonylag szűk szögtartományban láthatók. Az ilyen elvű színes képernyők előállítása költséges.

Ionizált gázos (plazma)képernyők

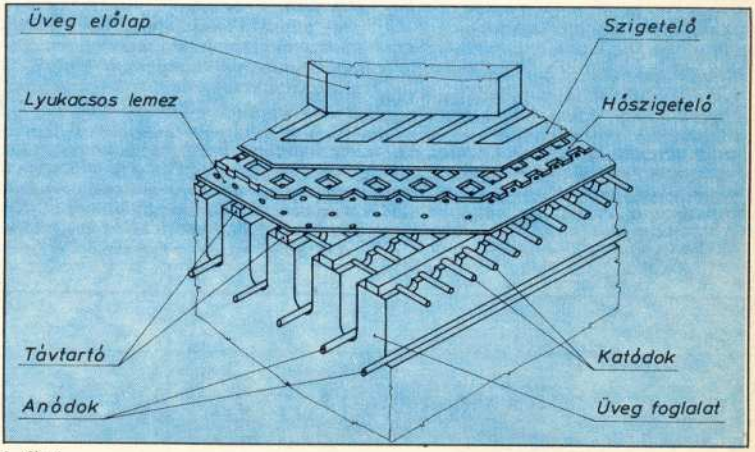
A plazmaképernyő fénycsőelv alapján működik, és mostoha körülmények között is kifogástalanul használható. Ezért főleg katonai területeken terjedt el, de a számítógépgyártó cégek már irodai alkalmazásra is gyártanak ilyen elven működő kijelzőket.

Működésük gázkisülési jelenségen alapul, amelyet a háló alakban elhelyezett anód és katód elektródák hoznak létre. Felget juttat a katódra és pozitívot az anódra, akkor azok keresztezési pontjain a kisülés következtében a gáz világító plazmává alakul. Ez a fényeffektus a homlokfelületen láthatóvá válik a lyukacsos lemez segítségével. A képfelbontás finomságát itt is az elektródaháló sűrűsége határozza meg.

A plazmaképernyők fényereje nagy, kontrasztgazdag képet adnak. Ezáltal a képernyőtartalom jól látható. A kép nyugodt, vibrációmentes. A technológiai fejlettség következtében ma már nagy felüle-



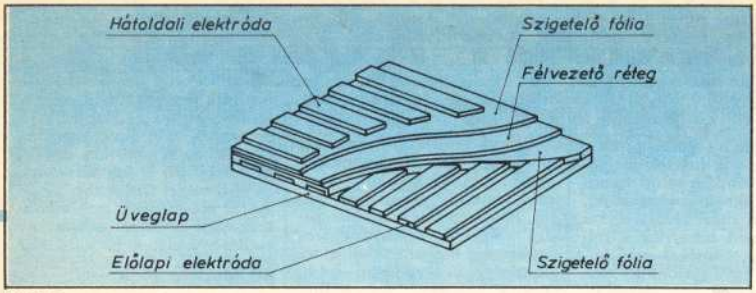
1. ábra



2. ábra

építésüket a *2. ábrán* láthatjuk. Az üveglapba hosszirányú hornyokat marnak, ebbe fektetik az anódot alkotó fémhuzalokat. Ezekre merőleges irányban helyezkednek el a katódszálak. A vezérlést az elektronika biztosítja. Az így elkészített hálót olyan lyukacsos lemez borítja, melynek furatai pontosan az anód-katód kereszteződési pontjai fölé esnek. Az elektródák közti teret nemesgázzal, például neonnal töltik ki. Ha a vezérlő elektronika negatív feszültségű, nagy felbontóképességű ernyőket készítenek. Ez a típusú képernyő viszonylag érzéketlen az ütéssel, rázkódással szemben.

Az aktív elvű megjelenítők között ez a technika a leggyakoribb. Ma már olyan képernyőket készítenek, amelyek 700 000 képpont megjelenítésére képesek. A megjelenítés színe fekete-fehér, helyesebben fekete-vörös. Színes megjelenítőt még nem sikerült ezzel a technikával előállítani. E megjelenítők hátránya még az is, hogy vi-



3. ábra

szonylag nagy energiaigényük miatt elemes üzemmódban nem használhatók.

Világító félvezetős (elektrolumineszcens) kijelzők

Az elektrolumineszcens kijelzők technikai szempontból közeli rokonai a plazmás kijelzőknek, azzal a különbséggel, hogy ez esetben nem ionizált gáz, hanem félvezető anyagból készült réteg világít.

Működésük meglehetősen egyszerű, felépítésüket a *3. ábra* mutatja. Itt nem gázkisülés világít, hanem az elektródák által keltett elektromos mezőben elhelyezkedő félvezető állítja elő a fényt. A plazmás kijelzőkhöz hasonlóan itt is mátrix alakban elhelyezkedő elektródákat találunk a képernyőfelület mögött kifeszítve úgy, hogy a függőleges és vízszintes elektródák között található az a „szendvics", amely két vékony szigetelőfólia között elhelyezkedő félvezető rétegből áll. A félvezető anyagtól függ az előállított fény színe.

Az elektrolumineszcens kijelzők fénye megfelel a hagyományos képcsövekének. Mivel a világító félvezető réteg szilárd, jobban viselik a rázkódást, ütközést. Rendkívül lapos kivitelben készíthetők, és kis súlyúak. Hátrányt jelent a viszonylag magas elektromosenergia-igény, így elemes üzemmód nem jöhet számításba. Meglehetősen drágák, egyelőre csak monokromatikus kép előállítására alkalmasak. Amerikai és japán kutatólaboratóriumok törekednek színes képernyő előállítására is.

Lapos képcsövek

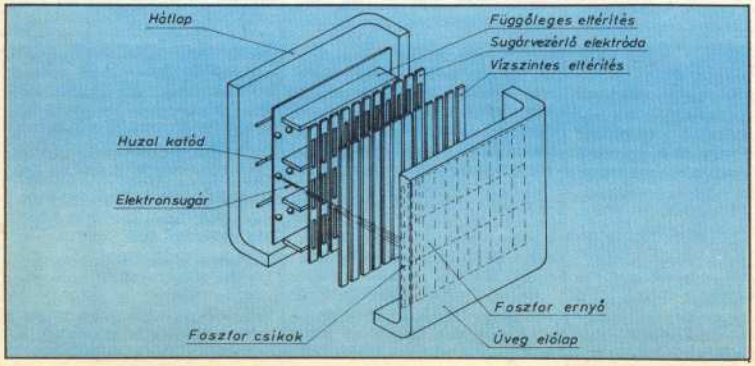
Ezek a kijelzők működési elvüket tekintve nem különböznek a hagyományos monitorokban alkalmazott képcsövektől. Fizikai megjelenésükben jóval laposabbak azoknál. A laposságot az biztosítja, hogy több katódot alkalmaznak. Felépítésüket a *4. ábra* szemlélteti.

Egy lapos, légüres térrel rendelkező „üvegkád" hátoldalán helyezkednek el a katódként szolgáló fémszálak. Ezek elé egy furattal ellátott maszkot helyezve, pontszerű elektronforrásokat nyerhetünk. Az elektronsugár intenzitását a maszk elé helyezett vezérlő elektródafelülettel szabályozhatjuk. A képcsőnek vízszintes és függőleges vezérlő elektródái is vannak, melyek meghatározzák, hogy a foszforbevonatú képernyő melyik képpontját kell kivilágítani. Így az egyes elektronsugarak csak a számukra kijelölt képernyőfelületet érhetik el. A fény így továbbra is a hagyományos foszforbevonatú ernyőn jelenik meg, az elektronok futási útja viszont jelentősen megrövidült, azaz a képernyő laposabbá vált.

Nagy előnye ennek a technikának, hogy a már ismert, hagyományos módon lehet színes képet előállítani. Hátránya a többi típussal szemben, hogy még mindig nem sikerült eléggé laposra tenni a képcsövet. Például a PANASONIC által ily módon előállított képcső 9 cm mély. További hátrány, hogy a képcsövek viszonylag súlyosak, nagy az energiaigényük, és ütésre ugyanúgy érzékenyek, mint régebbi társaik.

SZEBENSZKI SÁNDOR

4. ábra



ADOK – VESZEK – CSERÉLEK

Ebben a rovatban rövid, szöveges, a mikroszámítógépekkel kapcsolatos hirdetéseket közlünk. A díjszabás: közületeknek gépelt soronként (60 karakter) 100,— Ft, magánszemélyeknek az első sor 50,— Ft, minden további sor 20,— Ft. Az NJSZT tagjainak az első három sor ingyenes. Hirdetéseiket a szerkesztőség címére várjuk.

■ Commodore 16-os gépre írt játékprogramok cseréjéhez partnert keresek. László István, Békéscsaba, Máriássy u. 24.

■ Commodore VC20-ra készült programok eladása: assembler, disassembler, karaktermódosító, finomgrafikai, zene stb. Áruk: 100—300 Ft/db. A programok mindegyike alapgépen is használható. Kérjen tájékoztatót! Juhász György, 3100 Salgótarján, Pf. 157.

■ Timex Sinclair 1000 (ZX81) nagyon olcsón eladó. RAM: 2 kbájt. Telefon: 561-274 este.

■ Sinclair ZX81 16 k-s bővítővel, gépkönyvvel eladó. Lutár Zoltán, Budapest, Erzsébet u. 16. 1043. Telefon: 896-006.

■ VZ 200-as számítógéphez 16 kbájtos memóriabővítés eladó. Csáky Szabolcs, Dombóvár, Gyöngyvirág krt. 52.

■ Commodore 64-re készült játékprogramokat cserélek. Varga Vince, Szeged, Palánkai u. 18. Telefon: 19-637.

■ Commodore 64-re készült játékprogramokat cserélek. Wendler Gergely, Budapest, Krammer S. u. 24. IV. 25. Telefon: 582-462.

■ ZX—Spectrumhoz felhasználói és játékprogramokat cserélek. Válaszokat listával erre a címre kérek: Kiss László, Mezőtúr, Szabadság tér 29. 3/3/22. 5400.

■ 48 k-s Spectrumhoz felhasználói és játékprogramokat cserélek. Pillány Gábor, Pécs, Bajcsy-Zsilinszky út 3/9. 7622.

■ Eladó Commodore VIC—20, magyar nyelvű gépkönyvvel. Ára: 11 000 forint. Cím: Karcag, Kálvin út 19. 5300

■ VC20 datasettel, 20 k-s bővítővel és szakirodalommal eladó. Tóth Zoltán, Jászberény, Uszoda u. 1. 5100.

A MINŐSÉGÜGY KÖZÜGY

A végtermék minősége mint más jellemzők függvénye

Egyelőre tekintsünk el attól, hogy a minőséget egzaktságra törekvően meghatározzuk! (Ezt természetesen csak időlegesen tehetjük meg, hiszen például minden konkrét gyakorlati feladat tényleges megoldásában elkerülhetetlen az egzakt definiálás. Elgondolkoztató, hogy a gyakorlat a minőséget sokkal hamarabb és sokkal többször igyekszik pontosan mérhetően definiálni, mint ahogyan azt az elmélet teszi.)

Feltételezzük továbbá még azt is, hogy a minőséget mérni is tudjuk, és e mérés olyan, mintha eredménye egyetlen egyszerű egyenletes, monoton skálán való leolvasással lenne megállapítható; ugyanúgy, ahogy például tömeget mérünk.

Szedjük össze azokat a jellemzőket, amelyek a minőséget befolyásolják! E jellemzőket rögzített sorrendben egy vektorba fogva össze, a lehetséges vektorok halmazán egy minőség értékű (valós) függvényt értelmezhetünk. Természetes igény e függvény alakulásának, különösen szélső értékeinek megismerése. E feltérképező munka kielégítő elvégzése azonban általában nagyon nehéz, mert e függvények bonyolultak szoktak lenni. Mindig vannak azonban szerencsések. Sajnos általában kevesen. Keveseknek jut ugyanis az a szerencse, hogy a minőségfüggvény szélsőérték-meghatározási kísérletük kudarcba fullad. A többség ugyanis minden eszközzel kikényszeríti a sorstól a saját és vállalata balszerencséjét a szélső értékek kiszámítása — és „gyakorlati hasznosítása" — révén.

Nem kell túl sok gyakorlati tapasztalat ahhoz, hogy ezt belássuk. A minőséget ugyanis, mint más jellemzők függvényét vizsgálni szinte mindig egyenértékű egy katasztrofális félrecsúszással, azzal, hogy igyekeznek a legjobb minőséget megközelíteni. Pedig a minőség csak egy a jellemzők között, és közvetve vagy közvetlenül, szerepeltetni illik ugyan minden helyesen megállapított célfüggvényben (lásd később), de a minőséget a célfüggvénnyel azonosítani szinte mindig durva hiba.

És mégis csinálják. Az ilyen tevékenység káros volta csak akkor enyhül egy kicsit, ha a célfüggvénybe nem egy, hanem több lényeges minőségjellemzőt is bevesznek.

A helyes megoldás azonban csak úgy közelíthető meg, hogy minden lényeges jellemző szerephez jut a célfüggvényben. Hogy e jellemzők megfelelő szempontból mind minőségjellemzőknek foghatók fel, az egészen más kérdés. Nem volna semmi hiba, ha a célfüggvény a terméket más termékekkel együtt előállító gazdasági egység gazdálkodási minőségét mérné. Az ilyen felfogás — ez volna az igazi — azonban ma még erőltetettnek hatna, ezért kezeltük a minőségjellemzőket a „többi" jellemzőtől elkülönítetten.

Térjünk vissza a fogalmak tisztázására! A *célfüggvény* lényegében — röviden — olyan függvény, amely a választási lehetőségek közötti értékesség, fontosság (minőség!) szerinti megkülönböztetés elvégzését teszi lehetővé. A lehetőségek halmazán értelmezett, a lehetőségek értékét, fontosságát (minőségét!) mutató függvény.

A *minőség* pedig valamilyen meghatározott igényrendszer (kívánalomrendszer, elvárásrendszer) valami általi kielégítésének mértéke. A minőség jellemzője arról informál, hogy valami valamihez milyen mértékben illeszkedik. Minden minőségjellemző (minden minőség mérőszám), tehát szükségképp valamilyen approximáció jóságmérője — más nem lehet. Hogy sokféle követelményrendszer, sokféle ezt approximáló megvalósítás és sokféle jóságmérés lehetséges, az természetes. Ebből következőleg az is természetes, hogy e sokaságban megbízható elméleti eszközök nélkül könnyű eltévedni.

Hamis képet festenénk, ha azt állítanánk, hogy birtokunkban van legalábbis kielégítően jó módszer, vagy legalábbis nagyjából jól működő irányú — hacsak a józan kockázatvállalást és az okos óvatosságot nem tartjuk annak. Annak tartjuk, de hát ezeket nehéz lenne specifikus, tudományos eszköznek elfogadni. Talán nem is azok — többek annál.

Most, hogy a klasszikus értelemben vett minőséget — a minőség érdekében — eltávolítottuk a középpontból és egyrészt mint egy (vagy több) minőségjellemzőt bevontuk a célfüggvényben szereplő többi jellemző közé, másrészt pedig általánosítva a minőség fogalmát, minden jellemzőt minőségjellemzőként fogtunk fel, azzal a kérdéssel találjuk magunkat szemben, hogy hogyan határozzuk meg rendszerek viselkedésének minőségjellemzőit.

Erre a kérdésre — minden gyakorlati esetben alkalmazható — egzakt választ adni túlságosan merész vállalkozás volna. Konkrét válaszunkhoz pedig alapos helyzetfelmérés szükséges. Megállt tehát a tudomány? Többre nem futja, mint hibás előítéletek lerombolására? Részben ez a helyzet. Nem titkolhatjuk, hogy a minőséggel kapcsolatos bajok okai *nemcsak az emberi trehányságban és felelőtlenségben vannak*. A minőség-problematika megoldása nemcsak szigor és nemcsak morál kérdése. A minőséggel kapcsolatos mai válság más okai *a minőségfogalom és a minőségelmélet válságára mutatnak*. Baj van a minőségelmélet és alkalmazása minőségével is. A minőségelmélet jelenleg rendelkezésre álló módszerei gyártmány- és szolgáltatásminőséggel, ezek irányításával kapcsolatos nagyon egyszerű módszerek. Ezek pedig *nem alkalmasak* a jelenlegi nagy bonyolultságú és jellegében is más feladatok kezelésére és megoldására. Az, hogy ugyanazt a „minőség" szót használjuk ott is és itt is, csak megtévesztő — sokan be is dőlnek.

Néhány hasznos, konkrét munkára felkészítő gyakorlati tanáccsal fejezzük be mostani elmélkedésünket.

Hogyan fogjon hozzá a munkához a gyakorlati ember a „terepen", ha például kinevezik vállalati minőségőrnek vagy minőségfelelősnek, netán egy minőségügyi osztály együttesébe vagy élére kerülve. Játszva a lehetőségekkel, legyen szó például szoftverminőségről!

Mérje fel először a „terepet"! Állapítsa meg, hogy a jelenlegi termelési, szolgáltatási rendszerben milyen erők, milyen ösztönzések működnek, milyen irányban, és ezek milyen folyamatokat hoznak létre. Ha elér arra a szintre, hogy már nem csodálkozik azon, hogy a minőség miért olyan amilyen, elmondhatja, hogy az első lépést sikerrel tette meg. Ha még egy gráfot is fel tud rajzolni, amelyben a hatáskapcsolatokat legalább hozzávetőlegesen szemléltetni képes, bízhat benne, hogy kézben tudja majd tartani a további lépések irányítását is.

Ezek után következhet egy nehezebb feladat. Szerkesszen célfüggvényeket, amelyek a jelenlegi viszonyokra vonatkoznak! Ha most ez folyik, mire is törekszik most a rendszer? Mit tart jónak? Mit mivel szemben minősít saját számára értékesebbnek? Mi a fontos számára? Mi minél fontosabb a számára? Ha e feladat megoldásában nem is jut el kvantitatív eredményekig, a jó kvalitatív felmérésnek is rengeteg hasznát veheti.

Kellő jártasság után érdemes hozzáfogni olyan célfüggvények megszerkesztéséhez, amelyeket ideális esetben használnána. Különböző gazdasági, termelő, szolgáltató stb. egységeknek, funkcióknak természetesen külön-külön kell célfüggvényt szerkeszteni, hiszen az egész szervezet irányításában több „kisebb" irányítási tevékenység is szerepet játszik.

Az ilyen gyakorlatokat végző azonban nagy veszélyben van. Észre sem veszi, hogy valamiféle ideális eset, ideális állapot felé igyekszik terelni a befolyási körébe esőket, attól a veszedelmes és legtöbbször nagy kárt okozó előítélettől vezettetve, hogy az a legjobb, ha *a legjobbat* igyekszik megközelíteni. Ez a viselkedéspolitika azonban végzetesen hibás (is) lehet. Ha például már elkövettünk egy hibát, akkor nem mindig a hiba kijavítása a helyes politika. Ha valaki például rossz irányba indult el, és bizonyos idő múlva észrevette tévedését, nem biztos, hogy az lesz a legjobb, ha visszamegy oda, ahonnan elindult, és most másodszorra a helyes irányba indulva, megkezdi célja elérésére irányuló műveleteit. Ha közben a cél is változhat, az újrakezdő, hibakijavító politika helytelensége kiütközik.

Nem helyes tehát az a felfogás, amely általános ideális állapot elérésére törekszik, konkrét ideális viselkedés megvalósítása helyett. A minőségügy, különösen a mai gyorsan változó világban meg kell hogy szabaduljon a fél évszázaddal ezelőtti *állapotközpontúságtól*, amikor beállítottak és minél hosszabb ideig igyekeztek fenntartani egy jó állapotot. Ez akkor elég volt. Ma gyorsan változó körülmények között ez nemcsak lehetetlen, hanem *hibás* is volna. *Ma nem a létrehozás és fenntartás műveletei dominálnak, hanem a navigációé.* Ma navigálni kell.

És *nem a termék és szolgáltatás minőségén múlik a siker, hanem a navigáció minőségén*, amelybe természetesen a termék, szolgáltatás minősége is beleszól.

BASIC és gépi kód

Legutóbb a saját gépi kódú rutinok elhelyezésének lehetőségeiről és a gépi kódú eljárások aktivizálásának módjairól volt szó. Szerepelt egy kis feladat is: megfejteni, hogy mit csinál a példaprogramban szereplő gépi kódú rutin. A megoldás: megszámolja az A regiszter tartalmának 1-es értékű bitjeit, és a darabszámot kiírja a képernyőre.

Érdemes azon is gondolkodni, hogy hogyan oldható meg ez a feladat BASIC-ben?

Most a mikroprocesszor regisztereivel és néhány vezérlésátadó gépi utasítással ismerkedünk meg. Szükség lesz az áprilisi szám 23—26. oldalain található „A 65XX mikroprocesszor gépi és assembler szintű utasításkészlete" című táblázatra, melyre a továbbiakban „segédlet" néven fogok hivatkozni.

A regiszterek

A segédletben a Jelölések cím alatt találjuk a 65XX regisztereinek rövid leírását. Célszerű a táblázatban a regiszterek leírása és az SR bitjeinek leírása közé egy függőleges elválasztó vonalat húzni, hogy a két egymás melletti táblázat elkülönüljön egymástól.

Mikroprocesszorunknak 6 regisztere van. A memória egy bájtja nem regiszter, csak a jelölés egységessége miatt került a segédletben a regiszterek közé. A regiszterek általában 8 bitesek, kivétel a 16 bites utasításszámláló és a 9 bites veremmutató. Ez utóbbi nyolcbitesnek is tekinthető, hiszen a 9., a legmagasabb helyiértékű bit mindig 1-es.

Lássuk egyenként a regisztereket!

Az utasításszámláló

A program futása közben általában a következő végrehajtandó utasítás címét tartalmazza. Ez persze nem pontosan így igaz, működése sokkal bonyolultabb, de ebbe most korai lenne belemerülni. Azért 16 bites, hogy a 64 kilobájtos memória bármely címén levő utasítás elérhető legyen. Tartalma minden gépi utasítás végrehajtása közben változik, a vezérlésátadó utasítások tovább módosítják.

Angol nevét (program counter) PC-nek rövidítik. A továbbiakban mi is ezt a jelölést használjuk.

Az akkumulátor

A regiszternek is nevezik, általános célú regiszter. Az aritmetikai és logikai műveletek végrehajtása ebben a regiszterben történik. Bár a memóriamezők közötti adatátvitel az indexregisztereken keresztül is történhet, lehetőségei a különböző címzési módok révén, az A regiszteren keresztül a legnagyobbak. A processzor veremmemóriájához is csak ezen a regiszteren keresztül lehet közvetlenül hozzáférni.

Az indexregiszterek

Két indexregiszter van, az X és az Y. Az egyszerű indexelt címzési módoknál és a memóriabájtok közötti adatátvitelnél a szerepük azonos, de van néhány speciális feladat, melyekre csak az egyik indexregiszter alkalmas.

A veremmutató tartalmához való hozzáférésre és annak megváltoztatására csak az X regiszteren keresztül van lehetőség. Az úgynevezett indexelt indirekt címzési mód is csak ezzel a regiszterrel használható. (Ezzel a címzési móddal egyébként bemutató példánkon kívül eddig még nem találkoztam.)

A másik hasonló nevű az indirekt indexelt címzési mód. Ez az előbbivel ellentétben nagyon gyakran fordul elő, és csak az Y regiszterrel együtt használható.

A veremmutató

Az általunk használt mikroprocesszorok a memória $0100 és $01FF közötti területét veremmemóriaként használják. Erről később majd részletesebben írok. Most csak annyit, hogy a veremkezelő utasítások segítségével itt lehet bizonyos információkat átmenetileg tárolni. A veremmutató mindig a következő beírandó információ címét tartalmazza. Mivel az említett terület csak 9 bittel címezhető, a veremmutató regiszter 9 bites, de a legmagasabb helyiértékű bit — megváltoztathatatlanul — mindig 1-es értékű.

A gép bekapcsolása utáni első műveletek egyike a veremmutató inicializálása. Ez a verem legmagasabb címére való beállítást jelenti.

Az angol elnevezés (stack pointer) rövidítésével a veremmutatót SP-vel szokás jelölni.

Az állapotregiszter

Angol nevének rövidítése (processor status register) nyomán kétféle jelölés is használatos: P és SR. Bitjei, az úgynevezett jelzőbitek a segédlet SR BITJEI táblázatában találhatók. Az egyes bitek szerepéről majd az utasítások megismerésekor lesz szó, most csak kettőt említek.

Az 5. (nem használt) bitet a processzor minden gépi utasításnál 1-re állítja. Ennek nincs jelentősége, de az állapotregiszter tartalmának kiolvasásakor nem árt tudni róla.

A 3. (D) bit a decimális üzemmódot jelzi. A Commodore gépeken nem használják gépi szinten a decimális aritmetikát, ezért a bekapcsolás utáni első utasítások egyike a decimális mód letiltása. (Remélem, hogy e sorozat keretében sikerül a decimális aritmetika használatával megismerkednünk.)

A BASIC SYS utasításának használatakor a P regiszterhez hozzá tudunk férni. Erre a következő alkalommal egy gyakorlati példát is láthatunk.

Vezérlésátadó utasítások

Most néhány feltétel nélküli vezérlésátadó utasítást ismerünk meg, a következő alkalommal pedig már használni is fogjuk őket.

Közös tulajdonságaik:

— módosítják az utasításszámláló tartalmát,
— az állapotregiszter bitjeit nem változtatják,
— működésük összehasonlítható egy-egy BASIC utasításával.

A feltétel nélküli ugrás

A JMP utasítás két változata közül most csak az abszolút címzési módúval foglalkozunk. Gépi kódja: $4C.

A BASIC GOTO utasításhoz hasonlítható, de itt utasításssorszám helyett egy gépi cím az operandus. Az operandus kétbájtos, az ugrási cím bájtjai a már leírt módon alacsony helyiérték — magas helyiérték sorrendben követik egymást.

Ugrás szubrutinra

A JSR utasítás a GOSUB BASIC utasításhoz hasonlítható. Csak abszolút címzésű változata van. Gépi kódja: $20.

A JMP utasításhoz hasonlóan működik, de a szubrutinra ugrás előtt a verembe teszi a szubrutinból való visszatéréshez szükséges címet. A JSR utasítás kétbájtos operandusa a szubrutin címe.

Visszatérés szubrutinból

Az RTS utasításnak nincs operandusa. A RETURN BASIC utasításnak feleltethető meg. A veremből kiveszi a JSR által ott elhelyezett visszatérési címet, és az utasításszámláló regiszterbe teszi. A program futása a JSR utáni utasításon folytatódik. Gépi kódja: $60.

BARNA LÁSZLÓ

Z80 programozási gyakorlato

A sorozat mindazoknak szól, akik valamilyen könyvből már megtanulták, vagy éppen most tanulják a Z80 CPU programozását. Különböző gépfüggetlen példákon keresztül mutat be programozási fogásokat és ad az olvasó kezébe olyan kész szubrutinokat, amelyeket bátran használhat saját programjai részeként. Azt szeretnénk, hogy ha valaki a sorozatot végigolvassa, akkor bele tudjon vágni komolyabb problémákba, de tudja azt is, hogy nem kell mindent gépi kódban vagy assembly nyelven megírni.

Nézzük először, hogy mi is az assembly és a gépi kód között a különbség! A gépi kód a gép számára közvetlenül érthető, az assemblyt le kell fordítani gépi kódra. Általában gépi kódú programozásnak hívják azt, amikor az assembly programot kódtáblázat segítségével, kézzel fordítják. Fáraszító, sok hibalehetőséget magában rejtő munka. Egyesek képesek meglepően nagy programot így megírni, de erre az ember többnyire csak kényszerhelyzetben vállalkozik. Erre a munkára van ugyanis (ha van) egy segítőtárs: az assembler fordítóprogram, vagy röviden: assembler. Ez olyan program, amely a gépbe assembly nyelven beírt programot szintaktikusan ellenőrzi, és ha hibátlannak találja, akkor lefordítja és betölti a tárba.

Az assembler számára meg kell azt is mondanunk, hogy milyen módon fordítsa le a programot: milyen címtől kezdje a fordítást, hol hagyjon a program közben szabad helyet a tárban és mennyit stb. Ezekre pszeudo utasítások vagy más néven direktívák szolgálnak, melyek assemblerenként különbözőek, nagyon sokfélék lehetnek, ezért mindig utána kell nézni az általunk használt assembler leírásában. A sorokat megjelölhetjük címkékkel, amelyekhez a fordítás során a fordító azokat az értékeket rendeli, ahova az adott sorból generált kódnak az első bájtja kerül. Ez a használatban azt jelenti, hogy egy sor elé címkét írva, egy, a sorra hivatkozó CALL, JP vagy JR vagy akár egy LD utasítás után is elég leírni a címkét a szám helyett.

Lehetőség van arra is, hogy egy címkéhez általunk megadott értéket rendeljünk. Erre legtöbbször az EQU direktíva szolgál. Így például a CR EQU 13 utasítással a CR címkéhez (carrige return = kocsi vissza) 13-at rendelve, a program további részeiben mindenhova CR-t írhatunk, ahova 13-at írnánk. (Természetesen 136 helyett nem írhatunk CR6-ot!) Célszerű azonban csak olyan helyre írni ezt a címkét, ahol a 13 mint sorvégjel szerepel, egyébként, bár a program helyesen fordítódik le, nehezen olvasható, sőt félrevezető. Érdemes mindig olyan címkét használni, ami emlékeztet a jelentésre.

A program későbbi olvashatóságát segítik a program szövegében elhelyezett megjegyzések is, melyek a legtöbb assemblernél a programsor pontosvesszővel leválasztott végén vannak. Ezeket a fordító figyelmen kívül hagyja. Rendkívül fontosak, mert az assembly programok sokkal nehezebben olvashatók, mint a BASIC vagy PASCAL nyelven írtak.

Ugyancsak ezért nagyon fontos a programok dokumentálása. Assembly programozásnál a regiszterek nevei rögzítettek, és így nem utalnak arra, hogy mit tárolunk bennük. A regiszterek felhasználhatósága azonban bizonyos mértékig megszabja, hogy mire melyik regisztert használjuk. Így például a HL regiszterpárt arra szoktuk használni, hogy memóriahelyekre mutasson, a B regisztert ciklusszámlálónak, az A regisztert, azaz akkumulátort aritmetikai és logikai műveletekre és így tovább. Elég nehéz szabályt kimondani. Általánosan: egy-egy feladatra mindig azt a regisztert kell használni, amelyikkel a legrövidebben, leggyorsabban megoldható a probléma. Eleinte elég nehéz a választás, később azonban, bizonyos gyakorlatot szerezve, természetessé válik. Persze ekkor is megtörténhet, hogy egy rutin számára kényelmesebb, ha egy előzőleg megírt rutint átírunk úgy, hogy más regisztereket használunk, de ezt mindig jelezni kell a dokumentációban, illetve új dokumentációt kell készíteni. Egy-egy rutin dokumentációjának négy dolgot mindenképpen tartalmaznia kell:

1. Mit csinál a rutin.
2. A rutin mely regisztereket és memóriahelyeket használ fel bemenetként.
3. Mely regiszterekben, illetve memóriahelyeken adja az eredményt.
4. Mely egyéb regisztereket változtat meg.

Minden rutint meg lehet írni úgy, hogy azon regiszterek kivételével, amelyekben az eredményt adja, ne írjon át egy más regisztert sem, azaz hogy visszatéréskor e regiszterek értékei megegyezzenek a híváskor tartalmazottakkal. Erre többnyire verem- (stack-) kezelő utasításokat szoktunk használni. Ha tehát valamely szubrutinban használjuk a B és C regisztert, de meg akarjuk őrizni az értéküket a hívó program számára, akkor mielőtt felhasználnánk őket, egy PUSH BC utasítást kell végrehajtatni a géppel, majd a visszatérés előtt egy POP BC utasítást. Akkor is ki kell venni a veremből az értéket, ha rutin közben dől el, hogy még sincs szükség rá. Gyakori hibaforrás, hogy bent marad valami a veremben, például a PUSH BC-t nem követi sehol a program folyamán POP BC, és a RET utasítás így a BC verembe került értékét tölti a programszámlálóba, azaz ott folytatódik a futás, ez pedig többnyire elkerülendő. (Persze van kivétel is, a Spectrum gép BASIC értelmező programja ilyen módon hajtja végre az USR utasítást.)

Fontos tudni, hogy mely utasítások milyen jelzőbiteket (flageket) állítanak. Sokszor használják az akkumulátor törlésére az XOR A utasítást, ez azonban amellett, hogy nulla kerül az akkumulátorba, törli a carry (átvitel) bitet és egyre állítja a zéró jelzőbitet. Mivel a carry jelzőbit törlésére nincs közvetlen utasítás, csak az egymást követő SCF és CCF utasításokkal érhető el a carry törlése, ezért gyakran használják az AND A vagy OR A utasításokat a carry törlésére. Ez azonban nem mindig tehető meg, mert ezek az utasítások állítják a zéró jelzőbitet is A értékének megfelelően, ezért ugyanezen utasításokat szokták használni a CP 0, két bájtot elfoglaló utasítás helyett is. A zéróbitet egyre lehet állítani a CP A utasítással.

Ezek az apró programozói fogások rövidebbé és gyorsabbá teszik a programot. Igaz ugyan, hogy csak egy-két bájtról és milliszekundumokról van szó, de sok kicsi sokra megy. Konkrét példa: A HT–1080Z számítógép ROM programja eredetileg 8080 processzorra készült, így sok Z80 utasítást nem használtak fel a programozók. Nincs a ROM-ban mégsem olyan rutin, amelyet lényegesen rövidebben és gyorsabban meg lehetne írni, mégis egy, a teljes Z80 utasításkészletet használó BASIC értelmező mérhetően gyorsabban fut. Sokat lehet tanulni a profi programozók munkáiból. Kapható leírás, igaz nem kis pénzért, a Spectrum és a HT–1080Z számítógépek fixtáráról.

Nézzünk meg bevezetőnek három rövid példát.

1. Írjunk programot, amely HL címtől keres egy nem szóköz karaktert, azt betölti az akkumulátorba, HL-t pedig ráállítja a következő karakterre!

Először betöltjük A-ba a karaktert, mert egyrészt, ha szóköz, csak így tudjuk eldönteni, hogy valóban az-e, ha pedig nem, akkor egyébként is oda kell tenni. Növeljük HL-t, mert ha az akkumulátorban a szóköz kódja van, akkor úgyis a következő karaktert kell venni, ha pedig nem, akkor így HL a nem szóköz karakter utánra mutat. Ezután eldöntjük, hogy az akkumulátorban szóköz van-e, vagy sem. Ha az van, akor a hasonlító utasítás hatására zéró bit = 1, nem tér vissza a program, hanem veszi a következő karaktert. Ha szóköz van az akkumulátorban, akkor visszatér az őt hívó helyre *(1. program)*.

Gyakorlásképpen próbáljuk meg kézzel, kódtábla segítségével lefordítani a programot. Vajon mindig leáll ez a ciklus?

2. Írjunk szubrutint, amely kiszámítja az akkumulátorban ASCII kóddal adott hexadecimális számjegy értékét! Ha nem hexadecimális számjegy kódja van A-ban, akkor visszaérkezéskor carry = 1, különben carry = 0 legyen!

Ha „0" kódja és „9" kódja közé esik a számjegy, akkor „0" kódját kell levonni, ha „A" kódja és „F" kódja közé esik, akkor „A" kódja mínusz 10-et kell levonni ahhoz, hogy megkapjuk a számjegy értékét. Ha a kód nem tartozik egyik tartományba sem, akkor hibát kell jelezni *(2. program)*.

A kódból először levonjuk a „0" kódját. Ha ezután carry = 1, akkor az érték kisebb volt, mint „0" kódja, visszatérés, és carry jelzi a hibát. Ha a már kisebbített érték kisebb mint 10, azaz az eredeti érték nem volt nagyobb mint „9", akkor a 10-zel való hasonlítás után carry = 1 lesz, és ebben az esetben nem is kell tovább alakítanunk a számot, de mivel a hiba jelzése éppen a carry jelzőbittel történik, ezért az ellentettjére állítjuk. Ezután ha a szám nagyobb volt, mint „9" kódja, akkor levonunk belőle 7 = „A" – 10 – „0"-t. Így már megkaptuk a végső értéket, csupán azt kell eldönteni és a carry bitet aszerint beállítani, hogy az érték helyes-e (lásd az *1. ábrát*). Nem lehet az érték kisebb 10-nél és nagyobb 15-nél, hiszen a kód, ha helyes, „A" és „F" közé esik. Ha „0" és „9" közé esett, akkor a RET NC utasításnál megtörtént a visszatérés. Ha tehát a szám kisebb mint 10, a hasonlító utasítás után carry = 1, ha pedig nagyobb 15-nél, akkor a 16-tal való hasonlítás után carry = 0, amit az ellentettjére váltva, megkapjuk a hibajelzést.

Nézzük meg most ugyanezt a szubrutint egy másik assembly listán. Nem használunk megjegyzéseket, és nem használjuk az ASCII kódok karakterrel megadható alakját *(3. program)*.

Az így leírt program sokkal nehezebben olvasható. Az assembly nyelven megírt programnak soha nem szabad úgy kinéznie, mintha egy gépi kódról visszafordított lista lenne.

3. Írjunk szubrutint, amely összehasonlítja HL-t és DE-t! Az eredményt carry és zéróbitek jelezzék!

	CARRY	ZÉRÓ
HL < DE	1	0
HL = DE	0	1
HL > DE	0	0

1. ábra

2. ábra

Karakter	Kód	"0"-t levonva	"A"-10-"0"-t levonva
0	48	0	
1	49	1	
2	50	2	
3	51	3	
4	52	4	
5	53	5	
6	54	6	
7	55	7	
8	56	8	
9	57	9	
	58	10	3
	59	11	4
	60	12	5
	61	13	6
	62	14	7
	63	15	8
	64	16	9
A	65	17	10
B	66	18	11
C	67	19	12
D	68	20	13
E	69	21	14
F	70	22	15

A példa második mondatának értelme csupán az, hogy a carry és zéróbitek állapotából meghatározható legyen HL és DE viszonya. Lehet tehát, hogy zéró=1 azt jelentse: HL<DE. Ez azonban célszerűtlen, a szubrutint használó programot nehezen olvashatóvá teszi, sőt félrevezető is lehet, és a szubrutint is nehéz lenne így megírni, hosszú is, lassú is lenne. A legtermészetesebb az, ha az információt hasonlóképpen hordozzák a jelzőbitek, mint az egy bájtot hasonlító utasítások után (lásd a *2. ábrát* és a *4. programot*).

A szubrutin először csak a számok felső bájtját hasonlítja össze. Ha ezek nem egyenlőek, akkor az összehasonlítás eredményeképpen a carry bit E és L regiszterek értékétől függetlenül hordozza a megfelelő információt. Az alsó bájtokat csak akkor kell összehasonlítani, ha a felső bájtok egyenlőek.

Próbáljuk végig a program futását konkrét értékekkel! Mely regisztereket használja fel a rutin? Meg lehet-e őrizni A regiszter értékét PUSH AF és POP AF utasításokkal? Hogyan lehet megőrizni A regiszter értékét? Nézzük meg ennek a problémának egy másik megoldását! *(5. program)*

Az előző program hossza 6 bájt, ez csupán öt. Az felülírja az akkumulátort, ez nem. Az előzőnek azonban van egy nagy előnye, hogy lényegesen gyorsabb, még abban az esetben is, ha a felső bájtok egyenlőek. Hogy melyiket válasszuk, csak a konkrét helyzetben dönthetjük el, ha tudjuk, hogy mi a fontosabb: a sebesség vagy kis méret.

VERHÁS PÉTER

1. program

```
GTCHR LD A, (HL)
      INCL HL
      CP " "    ;szóköz kódja
      RET NZ
      JR GTCHR
```

2. program

```
HEXA SUB "0"   ;"0" kódja
     RET C   ;kisebb volt, mint "0"
     CP "9"-48+1   ;felső határ "9"
     CCF
     RET NC   ;nem volt nagyobb, mint "9"
     SUB "A"-10-"10"
     CP 10
     RET C   ;kisebb volt, mint "A"
     CP 16   ;felső határ "F" ellenőrzése
     CCF
     RET
```

3. program

```
     SUB 48
     RET C
     CP 10
     CCF
     RET NC
     SUB 7
     CP 10
     RET C
     CP 10H
     CCF
     RET
```

4. program

```
COMP LD A,H   ;felső helyiérték
     CP D     ;hasonlítás
     RET NZ   ;nem egyenlő készen is van
     LD A,L   ;alsó helyiérték
     CP E     ;hasonlítás
     RET
```

5. program

```
COMP PUSH HL     ;elmentés a kivonás miatt
     OR A        ;carry törlése
     SBC HL,DE   ;beállítjuk a jelzőbiteket
     POP HL      ;vissza az eredeti érték
     RET
```

Disassembler program Sinclair számítógépekre

PROGRAMOK

1982 nyarán kezdtem használni egy ZX81 számítógépet, ami akkor egészen új eszköz volt, és kezdetben teljesen elbűvölt. Néhány hónapig boldogan programoztam BASIC-ben, de egy idő múlva fúrni kezdte az oldalamat a kíváncsiság: hogyan is működik a számítógépem igazából, mi van a rejtélyes ROM-ban, amiről a Sinclair-kézikönyv nem túl sokat árul el.

Beszereztem tehát a megfelelő irodalmat a Z80 mikroprocesszor gépi kódú programozásáról, és megpróbáltam egyszerű programrészeket írni gépi kódban. Természetesen ez rendkívül fáradságos munka volt, mivel semmi szoftver nem segítette próbálkozásaimat. A legkisebb hiba esetén összeomlott a programom, és kezdhettem elölről az egészet. Rengeteg időm ráment, látható eredmény nélkül.

Végül megértettem, hogy nem jutok előbbre valamilyen szoftvertámogatás nélkül. Mivel abban az időben nem lehetett beszerezni ilyet, elhatároztam, hogy magam készítek egy olyan BASIC-ben írt programot, amely gépi kódú rutinokat fordít vissza assembly nyelvre, vagyis egy disassemblert.

Hogy miért éppen disassemblert és nem assemblert? Hát ehhez kitalálhatnék valamilyen körmönfont magyarázatot, de egyszerűbb, ha bevallom, hogy azért döntöttem így, mert ennek elkészítése sokkal egyszerűbb.

Nos, a disassembler elkészült, és sikeresen használtam hosszú ideig. A ZX81-en kicsit lassan működött, de sebaj! Amikor szereztem egy Spectrumot, azonnal átírtam arra is. Meglepetésemre a program a Spectrumon lényegesen gyorsabban működött!

Idővel a Sinclair gépek szoftverellátottsága gyökeresen megjavult. Ma tucatnyi jobbnál jobb assemblert és monitorprogramot lehet kapni, ezek nálunk is közismertek. E programok többségét gyors gépi kódban írták meg: az én egyszerű készítményem nem versenyezhet velük. Ma már én sem ezt használom programfejlesztéshez. Kitaláltam azonban egy olyan alkalmazást, amely ma is időszerűvé teszi. Nevezetesen megtanítottam a disassemblert arra, hogy felismerje a Spectrum ROM specialitásait: a táblázatokat, karaktergenerátort, üzeneteket és végül az RST 08 és RST 28H utasítások egyedi szintaktikáját.

Ian Logan magyarul „A Sinclair ZX-Spectrum ROM programja" címmel megjelent kitűnő könyve részletesen leírja a lebegőpontos aritmetika érdekes megvalósítását a Spectrumban. A kalkulátor rutinjait az RST 28H utasítást követő címkék hívják. Ezek úgy foghatók fel, mint egy programozható zsebszámológép egyszerű utasításai: tulajdonképpen egy egyszerűbb nyelv a bonyolultabb nyelven (BASIC) belül. Nos, a program most az RST 28H-t a Spectrumban használt címkékkel fordítja vissza.

A közreadott két program közül az egyik a program eredeti, ZX81-re készült változata *(1. lista)*. Ez mindkét gépen működik. Betöltése után mindkét gépen szükség van az adatok bevitelére is; ezek a mnemonikok, címek és egyebek. A program gyors működése érdekében ezeket az adatokat nem DATA-READ típusú programszerkezetekben, hanem tömbváltozókban tároljuk. Feltöltésük a 6000-től kezdődő sorokkal történik.

Először minden sorból töröljük ki a védelem céljából beírt REM kulcsszókat, majd GOTO 6000-rel indítsuk el a programot. Ekkor a program kérni fogja a megfelelő adatokat, amelyeket a *2., 3., 4. és 5. lista* szerint gondosan gépeljünk be. Ha valamelyik adatlistával végeztünk, célszerű megszakítani a program futását, visszairni az 1. lista szerint a lefutott programrészbe a REM-eket, majd GOTO-val újraindítva visszairatni az éppen bevitt listát. Ha hibát találunk, azt egyszerűen parancs üzemmódban javítsuk.

Fontos, hogy a programlistákon ne változtassunk, mivel a program kiszámított GOSUB utasításokkal működik. Ha tehát a sorszámok nem az 1. lista szerintiek, a program hibásan fog működni.

Ha az összes adatot hibátlanul bevittük, a 6000. sorszám feletti sorokat ki is törölhetjük. Vigyázni kell azonban, hogy CLEAR vagy RUN parancsot soha ne adjunk ki ezután, mivel ezek a változatokat és ezzel a bevitt adatokat is törlik: programunk nem fog működni.

Most programunk egyszerűbb változata működőképes. A ZX81-tulajdonosoknak ennyivel be kell érniök. A Spectrumosok, miután kazettára vagy microdrive-cartridge-ra mentették a programot, begépelhetik a *6. listán* levő másik programot, majd

1. lista

```
 100 CLS
 200 PRINT AT 4,10;"DISASSEMBLER
"
 210 PRINT TAB 10;"============"
 220 PRINT AT 8,5;"MINDEN SZÁM D
ECIMÁLIS!"
 230 PRINT AT 11,1;" ENTER  HASZNÁL
JON RUN VAGY CLEAR"
 240 PRINT AT 15,10;"PARANCSOT!"
 250 PRINT AT 18,2;"BREAK-KEL LE
HET MEGÁLLITANI"
 260 PRINT AT 21,0;"HELYIK CIMTO
L";
 270 INPUT J
 280 S
 290 LET I=0
 300 LET I=I+1
 340 LET K=1+PEEK J
 370 GO SUB (1000+2*VAL B$(K))
 380 LET J=J+L
 390 IF I<22 THEN GO TO 300
 400 GO SUB 400
 410 GO TO 200
 440 LET Q$=INKEY$
 450 IF Q$="Z" OR Q$="z" THEN CO
PY
 460 IF Q$="" THEN GO TO 440
 470 RETURN
 600 LET J=J+1
 610 LET A=PEEK J
 620 RETURN
 700 GO SUB 600
 710 IF A>127 THEN LET A=A
 720 IF A>143 THEN LET A=A-256
 740 RETURN
 800 GO SUB 600
 820 LET B=INT (A/16)
 830 LET C=A-B*16
 840 LET D$=B$(B+1)+B$(C+1)
 850 RETURN
1000 GO SUB 1300
1001 RETURN
1002 GO SUB 1500
1003 RETURN
1004 GO SUB 1700
1005 RETURN
1006 GO SUB 1700
1007 RETURN
1008 GO SUB 1800
1009 RETURN
1010 GO SUB 1900
1011 RETURN
1012 GO SUB 2100
1013 RETURN
1014 GO SUB 2500
1015 RETURN
1016 GO SUB 2300
1017 RETURN
1300 PRINT J;TAB 7;A$(K)
1301 RETURN
1500 LET X=VAL X$(K)
1502 IF X=1 THEN LET X=10
1503 PRINT J;TAB 7;A$(K, TO X);
1505 GO SUB 800
1510 PRINT A;A$(K,X+1 TO )
1520 RETURN
1700 LET X=VAL X$(K)
1702 IF X=1 THEN LET X=10
1704 PRINT J;TAB 7;A$(K, TO X);
1705 GO SUB 800
1710 LET G=A
1720 GO SUB 800
1730 LET G=A*256+G
1740 PRINT G;A$(K,X+1 TO );
1750 IF K=143 OR K=181 THEN PRIN
T
1760 PRINT
1780 RETURN
1800 LET X=VAL X$(K)
1802 IF X=1 THEN LET X=10
1803 LET Z=J
1804 PRINT J;TAB 7;A$(K, TO X);
1810 GO SUB 700
1820 LET A=A+2
1850 PRINT A;TAB 25;Z+A
1868 RETURN
1900 PRINT J;TAB 7;
1910 GO SUB 800
1920 LET AA=INT ((A+8)/8)-1
1924 LET BB=A-AA*8+1
1926 LET AA=AA+1
1928 PRINT E$(AA);" TO VAL P$(AA
);B$(BB)
1930 RETURN
2100 LET S$="IX"
2110 PRINT J;TAB 7;
2150 GO SUB 800
2160 LET M=CHR$ A
2140 FOR N=1 TO 40
2150 IF M$=R$(N,1) THEN GO TO 21
00
2160 NEXT N
2170 PRINT "DEFB ";A
2175 RETURN
2180 LET M=CODE R$(N,2)
2190 LET Q=CODE R$(N,3)
2200 GO SUB (3000+100*CODE R$(N,
1))
2210 RETURN
2300 LET S$="IY"
2310 GO TO 2110
2500 PRINT J;TAB 7;
2510 GO SUB 800
2520 LET M$=CHR$ A
2530 FOR N=72 TO 129
2540 IF M$=R$(N,1) THEN GO TO 25
00
2550 NEXT N
2560 PRINT "DEFB ";A
2570 RETURN
2580 LET M=CODE R$(N,2)
2590 GO SUB (4000+100*CODE R$(N,
1))
```

```
2600 RETURN
3100 PRINT Y$(N, TO M);S$;Y$(N,M
+1 TO )
3110 RETURN
3200 PRINT Y$(N, TO M);S$;Y$(N,M
+1 TO G);
3210 GO SUB 800
3220 LET G=A
3230 GO SUB 700
3240 PRINT A*256+G,Y$(N,G+1 TO )
3250 RETURN
3300 PRINT Y$(N, TO M);
3310 GO SUB 800
3320 LET G=A
3330 GO SUB 700
3340 PRINT A*256+G,Y$(N,M+1 TO G
);S$;Y$(N,G+1 TO )
3350 RETURN
3400 GO SUB 700
3410 PRINT Y$(N, TO M);S$;Y$(N,M
+1 TO G);A,Y$(N,G+1 TO )
3420 RETURN
3500 GO SUB 800
3510 PRINT Y$(N, TO M);S$;Y$(N,M
+1 TO );A
3600 GO SUB 800
3630 PRINT Y$(N,G+1 TO 7);A
3540 RETURN
3800 GO SUB 800
3610 LET G=A
3620 GO SUB 800
3630 LET M$=CHR$ A
3640 FOR N=1 TO 72
3650 IF M$=R$(N,1) THEN GO TO 36
90
3660 NEXT N
3670 PRINT "DEFB ";A
3680 RETURN
3690 LET M=CODE R$(N,2)
3700 LET Q=CODE R$(N,3)
3710 PRINT Y$(N, TO M);S$;Y$(N,M
+1 TO Q);Q,Y$(N,Q+1 TO )
3720 RETURN
4100 PRINT Y$(N)
4110 RETURN
4200 PRINT Y$(N, TO M);
4210 GO SUB 800
4220 LET U=A
4230 GO SUB 800
4240 LET U=A*256+U
4250 PRINT U;Y$(N,M+1 TO )
4260 RETURN
5700 GO TO 3300 REM
5790 PRINT J;CHR$ T
5990 GO TO 5990
6000 REM AZ ALABBI PROGRAMREszEK
AZ AdATTOMBOK FELTOLTESERE SZOLG
ALNAK, MUTTAK ELLENORIZTUK AZ A
PATOK HELYESSEGET. AZ 5990. SOR
FELETTI PROGRAMRESZ TOROLHETO.
6020 REM +++
6030 REM DIM A$(255,9)
6040 REM DIM B$(256)
6050 REM DIM R$(129,4)
6060 REM DIM X$(256)
6070 REM DIM E$(129,9)
6080 REM DIM S$(32,6)
6090 REM DIM T$(4)
6100 REM DIM P$(32)
6190 STOP
6200 REM +++
6210 CLS
6220 PRINT "I";TAB 4;"A$(I)";TAB
13;"X$(I)";TAB 19;"B$(I)"''''
6230 FOR I=1 TO 256
6240 PRINT I;TAB 1;
6250 REM INPUT A$(I)
6260 PRINT A$(I),TAB 15;
6270 REM INPUT X$(I)
6280 PRINT X$(I),TAB 21;
6290 REM INPUT B$(I)
6300 PRINT B$(I)
6310 NEXT I
6320 STOP
7000 REM +++
7010 CLS
7020 PRINT "I CODE R$(I,1,2,3,4)
-$(I)"
7030 FOR I=1 TO 129
7040 PRINT I;TAB 4;
7050 FOR J=1 TO 4
7060 REM INPUT C
7070 REM LET R$(I,J)=CHR$ C
7080 NEXT J
7090 REM INPUT Y$(I)
7100 FOR J=1 TO 4
7110 PRINT CODE R$(I,J);TAB (4+J
*4);
7120 NEXT J
7130 PRINT Y$(I)
7140 NEXT I
7150 STOP
7990 REM +++
8000 CLS
8005 PRINT "I  F$(I)"''''
8010 FOR I=1 TO 6
8020 PRINT I;TAB 4;
8030 REM INPUT F$(I)
8040 PRINT F$(I)
8050 NEXT I
8060 STOP
9100 REM +++
9110 CLS
9120 PRINT "I   E$(I)     P$(I)",,
,
9130 FOR I=1 TO 32
9140 PRINT I;TAB 4;
9150 REM INPUT E$(I)
9160 PRINT E$(I);TAB 15;
9170 REM LET P$(I)=STR$ P
9180 PRINT P$(I)
9190 NEXT I
9190 STOP
```

2. lista

```
I   A$(I)     X$(I)  B$(I)

1   NOP          3      0
2   LD BC,       6      2
3   LD (BC),A    9      0
4   INC BC       6      0
5   INC B        5      0
6   DEC B        5      0
7   LD B,        5      1
8   RLCA         4      0
9   EX AF,AF'    9      0
10  ADD HL,BC    9      0
11  LD A,(BC)    9      0
12  DEC BC       6      0
13  INC C        5      0
14  DEC C        5      0
15  LD C,        5      1
16  RRCA         4      0
17  DJNZ         5      4
18  LD DE,       6      2
19  LD (DE),A    9      0
20  INC DE       6      0
21  INC D        5      0
22  DEC D        5      0
23  LD D,        5      1
24  RLA          3      0
25  JR           3      4
26  ADD HL,DE    9      0
27  LD A,(DE)    9      0
28  DEC DE       6      0
29  INC E        5      0
30  DEC E        5      0
31  LD E,        5      1
32  RRA          3      0
33  JR NZ,       6      4
34  LD HL,       6      2
35  LD (),HL     4      3
36  INC HL       6      0
37  INC H        5      0
38  DEC H        5      0
39  LD H,        5      1
40  DAA          3      0
41  JR Z,        5      4
42  ADD HL,HL    9      0
43  LD HL,()     7      3
44  DEC HL       6      0
45  INC L        5      0
46  DEC L        5      0
47  LD L,        5      1
48  CPL          3      0
49  JR NC,       6      4
50  LD SP,       6      2
51  LD (),A      4      2
52  INC SP       6      0
53  INC (HL)     8      0
54  DEC (HL)     8      0
55  LD (HL),     8      1
56  SCF          3      0
57  JR C,        5      4
58  ADD HL,SP    9      0
59  LD A,()      6      3
60  DEC SP       6      0
61  INC A        5      0
62  DEC A        5      0
63  LD A,        5      1
64  CCF          3      0
65  LD B,B       6      0
66  LD B,C       6      0
67  LD B,D       6      0
68  LD B,E       6      0
69  LD B,H       6      0
70  LD B,L       6      0
71  LD B,(HL)    9      0
72  LD B,A       6      0
73  LD C,B       6      0
74  LD C,C       6      0
75  LD C,D       6      0
76  LD C,E       6      0
77  LD C,H       6      0
78  LD C,L       6      0
79  LD C,(HL)    9      0
80  LD C,A       6      0
81  LD D,B       6      0
82  LD D,C       6      0
83  LD D,D       6      0
84  LD D,E       6      0
85  LD D,H       6      0
86  LD D,L       6      0
87  LD D,(HL)    9      0
88  LD D,A       6      0
89  LD E,B       6      0
90  LD E,C       6      0
91  LD E,D       6      0
92  LD E,E       6      0
93  LD E,H       6      0
94  LD E,L       6      0
95  LD E,(HL)    9      0
96  LD E,A       6      0
97  LD H,B       6      0
98  LD H,C       6      0
99  LD H,D       6      0
100 LD H,E       6      0
101 LD H,H       6      0
102 LD H,L       6      0
103 LD H,(HL)    9      0
104 LD H,A       6      0
105 LD L,B       6      0
106 LD L,C       6      0
107 LD L,D       6      0
108 LD L,E       6      0
109 LD L,H       6      0
110 LD L,L       6      0
111 LD L,(HL)    9      0
112 LD L,A       6      0
113 LD (HL),B    9      0
114 LD (HL),C    9      0
115 LD (HL),D    9      0
116 LD (HL),E    9      0
117 LD (HL),H    9      0
```

feltölthetik adatokkal az előbb ismertetett módon, a *7. lista* szerint. Ezután a két programot MERGE paranccsal össze kell másolni, és ezzel a program végleges változata is használatra kész. Figyelem, még egyszer: csak GOTO 1-gyel indítható! A program kivitelét kazettára úgy, hogy betöltéskor automatikusan elinduljon, az olvasóra bízzuk.

A program beindulás után megkérdezi, melyik címtől kérjük a disassemblálást, majd elkezdi a fordítást. Az assembly lista a szabványos Z80 mnemonikokat írja ki. Relatív ugrásoknál a program a kiszámított abszolút címet is kiírja *(8. lista)*.

Olyan memóriaterületeken, ahol a Spectrum ROM-ban táblázatok vannak, a program DEFB „szám"-ot ír ki, ha a táblázat számokat tartalmaz, illetve karaktert, ha karaktert tartalmaz, például a kulcsszótáblázatban. A program minden számot decimális alakban jelez ki *(10. és 11. lista)*.

A karaktergenerátor memóriaterületén a program a címet és a pontraszter nyolcszorosára kinagyított képét írja ki. Így a képernyőre kirajzolódnak az egyes karakterek nagyított képei *(12. lista)*.

Az RST Ø8 utasítás a hibaüzeneteket kezeli. Minden RST Ø8 utasítást egybájtos szám követ, mely a megfelelő hibaüzenetet kódolja. Ezt a bájtot a program DEFB „szám"-ként disassemblálja.

Az RST 4Ø (vagy hexadecimális jelöléssel RST 28H) utasítást a Spectrum ROM a lebegőpontos kalkulátor belépési rutinjának használja. Az RST 4Ø utasítás után álló néhány bájt segítségével hívja a ROM a lebegőpontos kalkulátor egyes szubrutinjait. Így a CALL „cím" vagy a JP „cím" hárombájtos utasítások helyett csak egybájtosat használ. Logan említett könyve ezeknek a bájtoknak „beszédes" neveket: címkéket adott, mi is ezeket használtuk (lásd a 11. listát).

```
118 LD (HL),L     9     0
119 HALT          4     0
120 LD (HL),A     9     0
121 LD A,B        6     0
122 LD A,C        6     0
123 LD A,D        6     0
124 LD A,E        6     0
125 LD A,H        6     0
126 LD A,L        6     0
127 LD A,(HL)     9     0
128 LD A,A        6     0
129 ADD A,        6     0
130 ADD A,        6     0
131 ADD A,        6     0
132 ADD A,        6     0
133 ADD A,        6     0
134 ADD A,        6     0
135 ADD A,(HL     9     0
136 ADD A,        6     0
137 ADC A,        6     0
138 ADC A,        6     0
139 ADC A,        6     0
140 ADC A,        6     0
141 ADC A,        6     0
142 ADC A,        6     0
143 ADC A,(HL     9     0
144 ADC A,        6     0
145 SUB B         5     0
146 SUB C         5     0
147 SUB D         5     0
148 SUB E         5     0
149 SUB H         5     0
150 SUB L         5     0
151 SUB (HL)      8     0
152 SUB A         5     0
153 SBC A,B       7     0
154 SBC A,C       7     0
155 SBC A,D       7     0
156 SBC A,E       7     0
157 SBC A,H       7     0
158 SBC A,L       7     0
159 SBC A,(HL     1     0
160 SBC A,A       7     0
161 AND B         5     0
162 AND C         5     0
163 AND D         5     0
164 AND E         5     0
165 AND H         5     0
166 AND L         5     0
167 AND (HL)      8     0
168 AND A         5     0
169 XOR B         5     0
170 XOR C         5     0
171 XOR D         5     0
172 XOR E         5     0
173 XOR H         5     0
174 XOR L         5     0
175 XOR (HL)      8     0
176 XOR A         5     0
177 OR B          4     0
178 OR C          4     0
179 OR D          4     0
180 OR E          4     0
181 OR H          4     0
182 OR L          4     0
183 OR (HL)       7     0
184 OR A          4     0
185 CP B          4     0
186 CP C          4     0
187 CP D          4     0
188 CP E          4     0
189 CP H          4     0
190 CP L          4     0
191 CP (HL)       7     0
192 CP A          4     0
193 RET NZ        6     0
194 POP BC        6     0
195 JP NZ,        6     3
196 JP            3     3
197 CALL NZ,      8     3
198 PUSH BC       7     0
```

3. lista

```
199 ADD A,        6     1
200 RST 00        6     0
201 RET Z         5     0
202 RET           3     0
203 JP Z,         5     3
204 204           3     5
205 CALL Z,       7     3
206 CALL          5     3
207 ADC A,        6     1
208 RST 08        6     0
209 RET NC        5     0
210 POP DE        6     0
211 JP NC,        6     3
212 OUT (),A      5     1
213 CALL NC,      8     3
214 PUSH DE       7     0
215 SUB           4     1
216 RST 16        6     0
217 RET C         5     0
218 EXX           3     0
219 JP C,         5     3
220 IN A,()       6     1
221 CALL C,       7     3
222 222           3     6
223 SBC A,        6     1
224 RST 24        6     0
225 RET PO        6     0
226 POP HL        6     0
227 JP PO,        6     3
228 EX(SP),HL     9     0
229 CALL PO,      8     3
230 PUSH HL       7     0
231 AND           4     1
232 RST 32        6     0
233 RET PE        6     0
234 JP (HL)       7     0
235 JP PE,        6     3
236 EX DE,HL      8     0
237 CALL PE,      8     3
238 238           3     7
239 XOR           4     1
240 RST 40        6     0
241 RET P         5     0
242 POP AF        6     0
243 JP P,         5     3
244 DI            3     0
245 CALL P,       7     3
246 PUSH AF       7     0
247 OR            3     1
248 RST 48        6     0
249 RET M         5     0
250 LD SP,HL      8     0
251 JP M,         5     3
252 EI            2     0
253 CALL M,       7     3
254 254           3     8
255 CP            3     1
256 RST 56        6     0
```

```
1 CODE R$(1,1,2,3,4) Y$(1)

1    9    4    7    1    ADD ,BC
2    25   4    7    1    ADD ,DE
3    41   4    7    1    ADD ,HL
4    57   4    7    1    ADD ,SP
5    33   3    4    2    LD ,
6    34   4    6    3    LD (),
7    35   4    4    1    INC
8    42   3    5    2    LD ,()
9    43   4    4    1    DEC
10   52   5    6    4    INC (+)
11   53   5    6    4    DEC (+)
12   54   4    5    5    LD (+),
13   70   6    7    4    LD B,(+)
14   78   6    7    4    LD C,(+)
15   86   6    7    4    LD D,(+)
16   94   6    7    4    LD E,(+)
17   102  6    7    4    LD H,(+)
18   110  6    7    4    LD L,(+)
19   126  6    7    4    LD A,(+)
20   112  4    5    4    LD (+),B
21   113  4    5    4    LD (+),C
22   114  4    5    4    LD (+),D
23   115  4    5    4    LD (+),E
24   116  4    5    4    LD (+),H
25   117  4    5    4    LD (+),L
26   119  4    5    4    LD (+),A
27   134  7    8    4    ADD A,(+)
28   142  7    8    4    ADC C,(+)
29   150  5    6    4    SUB (+)
30   158  7    8    4    SBC A,(+)
31   166  5    6    4    AND (+)
32   174  5    6    4    XOR (+)
33   182  4    5    4    OR (+)
34   190  4    5    4    CP (+)
35   203  1    1    6
36   225  4    4    1    POP
37   227  8    8    1    EX (SP),
38   229  5    5    1    PUSH
39   233  4    4    1    JP ()
40   249  6    6    1    LD SP,
41   6    5    6    4    RLC (+)
42   14   5    6    4    RRC (+)
43   22   4    5    4    RL (+)
44   30   4    5    4    RR (+)
45   38   5    6    4    SLA (+)
46   46   5    6    4    SRA (+)
47   62   5    6    4    SRL (+)
48   70   7    8    4    BIT 0,(+)
49   78   7    8    4    BIT 1,(+)
50   86   7    8    4    BIT 2,(+)
51   94   7    8    4    BIT 3,(+)
52   102  7    8    4    BIT 4,(+)
53   110  7    8    4    BIT 5,(+)
54   118  7    8    4    BIT 6,(+)
55   126  7    8    4    BIT 7,(+)
56   134  7    8    4    RES 0,(+)
57   142  7    8    4    RES 1,(+)
58   150  7    8    4    RES 2,(+)
59   158  7    8    4    RES 3,(+)
60   166  7    8    4    RES 4,(+)
61   174  7    8    4    RES 5,(+)
62   182  7    8    4    RES 6,(+)
63   190  7    8    4    RES 7,(+)
64   198  7    8    4    SET 0,(+)
65   206  7    8    4    SET 1,(+)
66   214  7    8    4    SET 2,(+)
67   222  7    8    4    SET 3,(+)
68   230  7    8    4    SET 4,(+)
69   238  7    8    4    SET 5,(+)
70   246  7    8    4    SET 6,(+)
71   254  7    8    4    SET 7,(+)
72   64   0    0    1    IN B,(C)
73   72   0    0    1    IN C,(C)
74   80   0    0    1    IN D,(C)
```

```
75   88   0    0    1    IN E,(C)
76   96   0    0    1    IN H,(C)
77   104  0    0    1    IN L,(C)
78   120  0    0    1    IN A,(C)
79   65   0    0    1    OUT (C),B
80   73   0    0    1    OUT (C),C
81   81   0    0    1    OUT (C),D
82   89   0    0    1    OUT (C),E
83   97   0    0    1    OUT (C),H
84   105  0    0    1    OUT (C),L
85   121  0    0    1    OUT (C),A
86   66   0    0    1    SBC HL,BC
87   82   0    0    1    SBC HL,DE
88   98   0    0    1    SBC HL,HL
89   114  0    0    1    SBC HL,SP
90   67   4    0    2    LD (),BC
91   83   4    0    2    LD (),DE
92   99   4    0    2    LD (),HL
93   115  4    0    2    LD (),SP
94   6    0    0    1    NEG
95   69   0    0    1    RETN
96   70   0    0    1    IM 0
97   86   0    0    1    IM 1
98   94   0    0    1    IM 2
99   71   0    0    1    LD I,A
100  74   0    0    1    ADC HL,BC
101  90   0    0    1    ADC HL,DE
102  106  0    0    1    ADC HL,HL
103  122  0    0    1    ADC HL,SP
104  75   7    0    2    LD BC,()
105  91   7    0    2    LD DE,()
106  107  7    0    2    LD HL,()
107  123  7    0    2    LD SP,()
108  77   0    0    1    RETI
109  79   0    0    1    LD R,A
110  87   0    0    1    LD A,I
111  95   0    0    1    LD A,R
112  103  0    0    1    RRD
113  111  0    0    1    RLD
114  160  0    0    1    LDI
115  161  0    0    1    CPI
116  162  0    0    1    INI
117  163  0    0    1    OUTI
```

4. lista

```
118  168  0    0    1    LDD
119  169  0    0    1    CPD
120  170  0    0    1    IND
121  171  0    0    1    OUTD
122  176  0    0    1    LDIR
123  177  0    0    1    CPIR
124  178  0    0    1    INIR
125  179  0    0    1    OTIR
126  184  0    0    1    LDDR
127  185  0    0    1    CPDR
128  186  0    0    1    INDR
129  187  0    0    1    OTDR
```

A lebegőpontos kalkulátor működését egyszerűsítve úgy foghatjuk fel, hogy a ROM tartalmát általában Z80 gépi kódban írták meg, azonban az RST 40 utasítás után egy ennél magasabb szintű nyelv utasításainak értelmezésére tér át az interpreter. Ezeket az utasításokat egy „end calc" címkéjű utasítás zárja le. Ettől kezdve az interpreter visszatér a normál gépi kódú utasítások használatára.

A lebegőpontos kalkulátor néhány utasítása hasonlóan működik, mint a BASIC függvényei: független változókat, paramétereket vár. Ilyen például a „series-Ø6" vagy a „stk-data" utasítás. Ezek a paraméterek a Sinclair BASIC-nek megfelelő formátumú ötbájtos lebegőpontos számok vagy sztringek, melyek közvetlenül követik az illető címkét (utasítást). Ezeket a paramétereket a program DEFB „szám" alakban, bájtonként írja ki.

A disassembler program alapötlete rendkívül egyszerű. A program végtelen ciklusban minden bájtot sorban megvizsgál. Ha a K-adik bájt 203 (CBH) vagy 237 (EBH), akkor több-bájtos utasításról van szó, és a program elágazik. Ha nem, kinyomtatja az egybájtos mnemonikot, amely nem más, mint az A$ adattömb megfelelő eleme: A$ (PEEK K+1).

I	E$(I)	P$(I)
1	RLC	4
2	RRC	4
3	RL	3
4	RR	3
5	SLA	4
6	SRA	4
7	SRL	4
8	SRL	4
9	BIT 0,	6
10	BIT 1,	6
11	BIT 2,	6
12	BIT 3,	6
13	BIT 4,	6
14	BIT 5,	6
15	BIT 6,	6
16	BIT 7,	6
17	RES 0,	6
18	RES 1,	6
19	RES 2,	6
20	RES 3,	6
21	RES 4,	6
22	RES 5,	6
23	RES 6,	6
24	RES 7,	6
25	SET 0,	6
26	SET 1,	6
27	SET 2,	6
28	SET 3,	6
29	SET 4,	6
30	SET 5,	6
31	SET 6,	6
32	SET 7,	6

5. lista

I	C$(I)		
		41	sqr
		42	sgn
1	jump-true	43	abs
2	exchange	44	peek
3	delete	45	in
4	subtract	46	usr-no
5	multiply	47	str$
6	division	48	chr$
7	to-power	49	not
8	or	50	duplicate
9	no-&-no	51	n-mod-m
10	no-l-eql	52	jump
11	no-gr-eql	53	stk-data
12	nos-neql	54	dec-jr-nz
13	no-grtr	55	less-0
14	no-less	56	greater-0
15	nos-eql	57	end-calc
16	addition	58	get-argt
17	str-&-no	59	truncate
18	str-l-eql	60	fp-calc-2
19	str-gr-eq	61	e-to-fp
20	strs-neql	62	re-stack
21	str-grtr	63	series-06
22	str-less	64	series-08
23	strs-eql	65	series-0C
24	strs-add	66	stk-zero
25	val$	67	stk-one
26	usr-$	68	stk-half
27	read-in	69	stk-pi/2
28	negate	70	stk-ten
29	code	71	st-mem-0
30	val	72	st-mem-1
31	len	73	st-mem-2
32	sin	74	st-mem-3
33	cos	75	st-mem-4
34	tan	76	st-mem-5
35	asn	77	get-mem-0
36	acs	78	get-mem-1
37	atn	79	get-mem-2
38	ln	80	get-mem-3
39	exp	81	get-mem-4
40	int	82	get-mem-5

7. lista

8. lista

9. lista

10. lista

11. lista

12. lista

6. lista

```
  10 REM SAVE "DIS" LINE 1
 110 LET U=0
 345 IF (J)=149 AND J(=653) OR (
J)=1134 AND J(=1217) OR (J)=2465
 AND J(=2547) OR (J)=2577 AND J(
=2594) THEN GO TO 5000
 350 IF (J)=4000 AND J(=4006) OR
 (J)=5009 AND J(=5460) OR (J)=55
51 AND J(=5571) OR (J)=5574 AND
J(=5587) OR (J)=5677 AND J(=5683
) THEN GO TO 5000
 355 IF (J)=5910 AND J(=5915) OR
 (J)=6010 AND J(=6016) OR (J)=67
28 AND J(=6934) OR (J)=7169 AND
J(=7180) THEN GO TO 5000
 360 IF (J)=9522 AND J(=9646) OR
 (J)=10133 AND J(=10172) OR (J)=
12997 AND J(=13146) OR (J)=15616
 AND J(=16383) THEN GO TO 5000
 365 IF U=J THEN GO SUB 1000: GO
 TO 380
 420 PRINT #1;"Z-COPY, barmely m
as gomb-tovabb"
 430 BEEP .4,7
1300 IF J=U THEN GO TO 1338
1310 IF K=240 THEN GO TO 1330
1315 IF K=208 THEN GO TO 1450
1330 LET QQ=0
1335 LET J=J+1: LET I=I+1
1338 K=PEEK J+1
1340 IF QQ=1 THEN LET U=J+K-1: P
RINT J;TAB 9;"DEFB ";K-1: LET QQ
=0: GO TO 1370
1341 IF K=135 THEN LET K=63: GO
TO 1350
1342 IF K=137 THEN LET K=64: GO
TO 1350
1343 IF K=141 THEN LET K=65: GO
TO 1350
1344 IF K)=160 AND K(=165 THEN L
ET K=K-95: GO TO 1350
1345 IF K)=192 AND K(=198 THEN L
ET K=K-122: GO TO 1350
1346 IF K)=224 THEN LET K=K-148
1348 IF K=92 THEN PRINT J;TAB 7;
"RST 40": GO TO 1370
1349 IF K)82 THEN GO TO 1305
1350 PRINT J;TAB 9;C$(K)
1351 IF K=53 THEN GO SUB 1410
1352 IF K=63 THEN FOR U=1 TO 6:
GO SUB 1410: NEXT U
1353 IF K=64 THEN FOR U=1 TO 8:
GO SUB 1410: NEXT U
1354 IF K=65 THEN FOR U=1 TO 12:
 GO SUB 1410: NEXT U
1355 IF K=1 OR K=52 OR K=54 THEN
 LET QQ=1
1360 IF K=57 THEN RETURN
1370 IF I(22 THEN GO TO 1335
```

```
1380 GO SUB 420: CLS : LET I=0:
GO TO 1330
1400 RETURN
1410 FOR L=1 TO 5: LET J=J+1: LE
T I=I+1: PRINT J;TAB 7;"DEFB ";P
EEK J;TAB 18;CHR$ ((L=1)*48+32):
 IF I(22 THEN GO TO 1440
1420 GO SUB 420: CLS : LET I=0
1440 NEXT L: RETURN
1450 LET I=I+1: LET J=J+1
1460 PRINT J;TAB 7;"DEFB ";PEEK
J: RETURN
1770 IF K=206 AND Q=13154 THEN G
O TO 1330
5000 LET Y=0: LET T=PEEK J
5010 IF J)516 THEN GO TO 5050
5020 GO TO 5700
5050 IF J)653 THEN GO TO 5080
5060 GO TO 5790
5080 IF J)1219 THEN GO TO 5110
5100 GO TO 5750
5110 IF J)2547 THEN GO TO 5160
5120 IF J=2465 THEN GO TO 5750
5130 LET T$=CHR$ T: IF T=13 THEN
 LET T$="CR"
5150 GO TO 5190
5160 IF J)4008 THEN GO TO 5190
5180 GO TO 5750
5190 IF J)5461 THEN GO TO 5210
5200 GO TO 5700
5210 IF J)13145 THEN GO TO 5230
5220 GO TO 5750
5230 LET R=.501: PRINT J;TAB 9;T
;TAB 13;
5240 LET T$="          ": FOR L=8
TO 1 STEP -1
5250 LET Q=INT (T*R): LET T=T-Q*2
5260 IF T=0 THEN LET T$(L)="·"
5270 IF T=1 THEN LET T$(L)="■"
5280 LET T=Q: NEXT L
5290 PRINT T$
5300 GO TO 380
5700 IF T)127 THEN LET T=T-128:
LET Y=1
5710 PRINT J; INVERSE Y;CHR$ T
5720 GO TO 380
5750 PRINT J;TAB 9;"DEFB ";T
5760 GO TO 380
5790 PRINT J;CHR$ T
5800 GO TO 380
9110 REM DIM C$(82,9)
9200 REM ***
9210 CLS
9220 PRINT "I   C$(I)",,,,
9230 FOR I=1 TO 82
9240 REM I;TAB 4;
9250 REM INPUT C$(I)
9260 PRINT C$(I)
9270 NEXT I
```

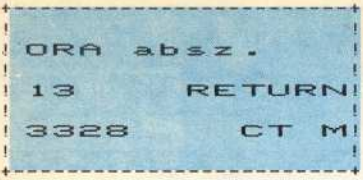
Ha a K-adik bájt nem 203 vagy 237, akkor az utasítás több-bájtos, a program az utasítás második bájtjából kiszámítja, melyik BASIC sorban van az a szubrutin, amelyik az adott utasítást visszafordítja assembly nyelvre. Ez a program 37Ø-es sorában történik. Az említett végtelen ciklus pedig a 28Ø-as és 41Ø-es BASIC sorok között található.

A több-bájtos utasítások fordítása az egybájtos utasításokéhoz hasonlóan kezdődik. Néhány utasítás szintaktikája azonban lényegesen bonyolultabb. A program alapelve az, hogy az időt rabló IF-THEN szerkezeteket lehetőség szerint kerüljük el. Ezt az elvet azonban nem tartottam be következetesen: a ritkábban előforduló Z80 utasításokat a program a szokott módon keresi és fejti meg. Ez a következetlenség jelentősen csökkenti a program terjedelmét.

Ismeretes, hogy a disassemblálás nem egyértelmű művelet. A visszafordítandó kódlistán kívül ismernünk kell a kezdőcímet és azt, hogy vajon a kódlista valóban programot testesít meg, vagy valamilyen más numerikus vagy szöveges adatot jelent. Másrészről léteznek olyan kódkombinációk, amelyek nem felelnek meg semmilyen assembly utasításnak sem. Ilyen esetben a program az értelmezhetetlen bájtokat DEFB „szám"-ként fordítja.

A program könnyen változtatható. Így például fel lehet váltani a decimális kijelzést hexadecimálisra, oktálisra vagy binárisra, vagy be lehet építeni az interaktív választást e lehetőségek között.

DR. KABOLDY PÉTER



Commodore 64 segédtáblázat

Ismert dolog, hogy a Commodore 64 számítógépek BASIC-je nem nyújt elég támogatást a gép hardverlehetőségeinek teljes kihasználásához. Ezért sokan alkalmazzák azt a módszert, hogy BASIC nyelven megírt főprogramjukból gépi kódú szubrutinokat hívnak, amelyek a tár különböző részein helyezkednek el. Leggyakrabban a kazettapuffert (helye a tárban: 828—1019 vagy hexadecimálisan $033C—$03FB), illetve a 49152 ($C000)-nál kezdődő 4 k-s RAM-területet használják erre a célra. De hogyan kerülnek a gépi kódú rutinok a megfelelő helyre? Erre több lehetőség is van; például monitorprogram segítségével, amely megengedi az utasítások mnemonikus formában történő bevitelét. Az így megírt program lemezre menthető, majd a főprogram indítása előtt a megfelelő tárcímre tölthető.

Nem mindig célszerű azonban ezt a megoldást választani, hiszen ha a program csak néhány utasítást tartalmaz, akkor fölösleges háttértároló segítségét igénybe venni. Ebben az esetben a legegyszerűbb az utasításkódoknak megfelelő számokat DATA-sorokban tárolni, majd READ utasításokkal a kívánt tárcímre olvastatni.

Többek között ehhez a megoldáshoz nyújthat hasznos segítséget a *táblázat*, amely első pillantásra kissé zsúfoltnak tűnhet, használata azonban rövid idő alatt elsajátítható.

Válasszunk ki egy tetszőleges mezőt és nézzük meg, milyen információkat tartalmaz! Legyen ez a 0. sor és a D. oszlop találkozásában található, azaz a $00 hexadecimális számnak megfelelő mező (lásd az *ábrát*).

A felső sorban található „ORA absz." a gépi kódú programozásban jártasak számára magától értetődő, ez a $0D utasításkódnak megfelelő gépi kódú utasítás mnemonikus formája a megfelelő címzésmóddal együtt. Az adott esetben abszolút címzésmódú utasításról van szó.

A középső sorban két információ is található: a 13 a $0D hexadecimális számnak megfelelő decimális érték; a RETURN a PRINT CHR$(13) utasítás hatását mutatja. Általánosítva: a bal oldali szám a jobboldalt látható vezérlőutasítás, karakter stb. ASCII kódja. Ha a vizsgált mező jobb alsó sarkában is található valamilyen információ (az adott esetben CT M), akkor ez azt jelenti, hogy az adott funkció még egy módon megvalósítható, példánk esetén a CTRL és az M billentyűk együttes lenyomásával.

Végül a bal alsó sarokban látható számról: ez mindig a fölötte levő szám 256-szorosa, azaz 13*256=3328.

Nézzünk néhány példát a táblázat gyakorlati alkalmazására! Legyen az első feladat a $C01B hexadecimális szám átváltása decimálisra! Első lépésként válasszuk szét az adott számot a felső (HIGH) és alsó (LOW) bájtokra! Keressük ki a táblázatból a felső bájtnak ($C0) megfelelő decimális értéket, amely természetesen az adott mező bal alsó sarkában levő 256-szoros számot jelenti. Ez a mi esetünkben 49 152. Végezzük el ugyanezt az alsó bájttal ($1B) is, itt azonban a középső sorban levő számra lesz szükségünk, azaz 27-re. Adjuk össze a két számot. Ezzel a feladatot meg is oldottuk: a $C01B hexadecimális szám decimális megfelelője tehát 49 152+27=49 179.

Visszafelé majdnem ugyanilyen egyszerű a számolás. Váltsuk át hexadecimálisra a 34 519-et! Keressük ki a táblázatból az adott számnál nem nagyobb, hozzá legközelebb eső számot! Ez a mi esetünkben 34 304, ezzel megkaptuk a felső bájtot, azaz $86-ot. Vonjuk ki ezt a számot az eredeti számból: 34 519−34 304=215, az ennek megfelelő hexadecimális szám ($D7) adja az alsó bájtot. Eredményünk tehát: 34 519=$86D7.

Harmadik példánk azt mutatja meg, hogy hogyan lehet a legkényelmesebben letiltani, illetve engedélyezni a gép kétféle karakterkészlete közötti átkapcsolást. Bizonyára sokak előtt ismert, hogy a PRINT CHR$(8) utasítás letiltja a C= és a SHIFT billentyűk egyidejű lenyomásával végrehajtható átváltást, a PRINT CHR$(9) utasítás pedig újra engedélyezi az eredeti állapotot. Ha azonban megnézzük táblázatunkban a $08 és a $09 mezőket, akkor észrevehetjük, hogy ugyanezt a hatást sokkal egyszerűbben is elérhetjük a CTRL és a H, illetve a CTRL és az I billentyűk egyidejű megnyomásával.

Végül nézzük egy rövid gépi kódú program DATA sorokba írását a táblázat segítségével. Legyen a feladat egy fekete A betű kiírása a képernyő bal felső sarkába! Először írjuk meg a forrásnyelvi programot. Ehhez azonban ismernünk kell néhány címet és kódot. Az A betű képernyőkódja 1, a fekete színkódja 0. A képernyőtároló első bájtjának címe, ahová a betű kerül: 1024, a színmemória megfelelő címe, ahová majd a fekete színkódja kerül: 55 296. Keressük ki a már leírt eljárás segítségével ezeknek a számoknak a hexadecimális megfelelőjét, majd írjuk meg a programot!

```
LDA #$01
STA $0400
LDA #$00
STA $D800
RTS
```

Vegyük sorra az utasításokat és a hozzájuk tartozó operandusokat! Az LDA # utasítás kódja a táblázat alapján $A9 (=169), az operandus $01 (=1). A következő utasítás abszolút címzésmódú STA kódja $8D (=141). Miután itt 2 bájtos az operandus, a bájtok felcserélve kerülnek a tárgyprogramba, tehát $00 (=0);$01 (=1) sorrendben! A továbbiakban ennek megfelelően folytatva, végezetül a kész DATA-sor:

```
DATA 169,1,141,0,4,169,0,141,0,218,96
```

Azoknak, akik már jártasak a gépi kódú programozásban, bizonyára feltűnt, hogy olyan mnemonikok is találhatók a táblázatban (ezek eltérő színháttérrel vannak jelölve), amelyek ismeretlenek előttük. A Mikroszámítógép Magazin egyik régebbi száma közölt egy cikket „Ismeretlen kódok" címmel, amelyben ezek egy részéről már volt szó, csak az elnevezések tértek el az itt leírtaktól. Az itt közölt táblázat az SMON nevű monitorprogram elnevezéseit használja, és csekély kivételtől eltekintve szinte az összes lehetséges kódot értelmezi: az elméletileg lehetséges 256-ból 242-t! Miután ezeknek az utasításoknak a többsége valószínűleg ismeretlen az olvasó előtt, röviden nézzük át a kódok jelentését!

A legtöbb „új" utasítás tulajdonképpen két másik, már ismert utasítást egyesít, tehát a státuszregiszter megfelelő jelzőbitjeit is ennek függvényében állítja.

LAX	Load Akku and X Végrehajt egy LDA és egy LDX utasítást a címzésmódnak megfelelően
DCP	Decrement and ComPare DEC és CMP utasítások egyesítése
ISC	Increment and SubtraCt INC és SBC
RLA	Rotate Left AND Akku ROL és AND
RRA	Rotate Right and Add with carry ROR és ADC
SLO	Shift Left OR Akku

	0	1	2	3	4	5	6	7
0	BRK 0 0	ORA (zp,x) 1 CTRL A 256	CRA 2 CTRL B 512	SLO (zp,x) 3 CTRL C 768	NOP-2 4 CTRL D 1024	ORA zp 5 FEHÉR 1280 CTRL E	ASL zp 6 CTRL F 1536	SLO zp 7 CTR 1792
1	BPL 16 CTRL P 4096	ORA (zp),y 17 CRSR ↕ 4352 CTRL Q	CRA 18 RVS ON 4608 CTRL R	SLO (zp),y 19 HOME 4864 CTRL S	NOP-2 20 DEL 5120 CTRL T	ORA zp,x 21 CTRL U 5376	ASL zp,x 22 CTRL V 5632	SLO zp 23 CTR 5888
2	JSR absz. 32 SPACE 8192	AND (zp,x) 33 ! 8448	CRA 34 " 8704	RLA (zp,x) 35 # 8960	BIT zp 36 $ 9216	AND zp 37 % 9472	ROL zp 38 & 9728	RLA zp 39 9984
3	BMI 48 0 12288	AND (zp),y 49 1 12544	CRA 50 2 12800	RLA (zp),y 51 3 13056	NOP-2 52 4 13312	AND zp,x 53 5 13568	ROL zp,x 54 6 13824	RLA zp 55 14080
4	RTI 64 @ 16384	EOR (zp,x) 65 A 16640	CRA 66 B 16896	SRE (zp,x) 67 C 17152	NOP-2 68 D 17408	EOR zp 69 E 17664	LSR zp 70 F 17920	SRE zp 71 18176
5	BVC 80 P 20480	EOR (zp),y 81 Q 20736	CRA 82 R 20992	SRE (zp),y 83 S 21248	NOP-2 84 T 21504	EOR zp,x 85 U 21760	LSR zp,x 86 V 22016	SRE zp, 87 22272
6	RTS 96 24576	ADC (zp,x) 97 24832	CRA 98 25088	RRA (zp,x) 99 25344	NOP-2 100 25600	ADC zp 101 25856	ROR zp 102 26112	RRA zp 103 26368
7	BVS 112 28672	ADC (zp),y 113 28928	CRA 114 29184	RRA (zp),y 115 29440	NOP-2 116 29696	ADC zp,x 117 29952	ROR zp,x 118 30208	RRA zp, 119 30464
8	NOP-2 128 32768	STA (zp,x) 129 NARANCS 33024	NOP-2 130 33280	SAX (zp,x) 131 33536	STY zp 132 33792	STA zp 133 f1 34048	STX zp 134 f3 34304	SAX zp 135 34560
9	BCC 144 FEKETE 36864	STA (zp),x 145 CRSR ↑ 37120	CRA 146 RVS OFF 37376	— 147 CLR 37632	STY zp,x 148 INST 37888	STA zp,x 149 BARNA 38144	STX zp,y 150 ROZSASZ 38400	SAX zp 151 S.S 38656
A	LDY # 160 SH-SPACE 40960	LDA (zp,x) 161 41216	LDX # 162 41472	LAX (zp,x) 163 41728	LDY zp 164 41984	LDA zp 165 42240	LDX zp 166 42496	LAX zp 167 42752
B	BCS 176 45056	LDA (zp),y 177 45312	CRA 178 45568	LAX (zp),y 179 45824	LDY zp,x 180 46080	LDA zp,x 181 46336	LDX zp,y 182 46592	LAX zp 183 46848
C	CPY # 192 49152	CMP (zp,x) 193 49408	NOP-2 194 49664	DCP (zp,x) 195 49920	CPY zp 196 50176	CMP zp 197 50432	DEC zp 198 50688	DCP zp 199 50944
D	BNE 208 53248	CMP (zp),y 209 53504	CRA 210 53760	DCP (zp),y 211 54016	NOP-2 212 54272	CMP zp,x 213 54528	DEC zp,x 214 54784	DCP zp 215 55040
E	CPX # 224 SH-SPACE 57344	SBC (zp,x) 225 57600	NOP-2 226 57856	ISC (zp,x) 227 58112	CPX zp 228 58368	SBC zp 229 58624	INC zp 230 58880	ISC zp 231 59136
F	BEQ 240 61440	SBC (zp),y 241 61696	CRA 242 61952	ISC (zp),y 243 62208	NOP-2 244 62464	SBC zp,x 245 62720	INC zp,x 246 62976	ISC zp 247 63232

	8	9	A	B	C	D	E	F
	PHP 8 SHIFT-C-TILT 2048 CTRL H	ORA # 9 SHIFT-C-ENGED 2304 CTRL I	ASL A 10 CTRL J 2560	— 11 CTRL K 2816	NOP-3 12 CTRL L 3072	ORA absz. 13 RETURN 3328 CTRL M	ASL absz. 14 KISBETŰ 3584 CTRL N	SLO absz 15 CTRL O 3840
	CLC 24 CTRL X 6144	ORA absz,y 25 CTRL Y 6400	NOP 26 CTRL Z 6656	SLO absz,y 27 CTRL [6912	NOP-3 28 PIROS 7168	ORA absz,x 29 CRSR→ 7424	ASL absz,x 30 ZÖLD 7680	SLO absz,x 31 KÉK 7936
	PLP 40 (10240	AND # 41) 10496	ROL A 42 ★ 10752	— 43 + 11008	BIT absz. 44 ' 11264	AND absz. 45 — 11520	ROL absz. 46 . 11776	RLA absz 47 / 12032
	SEC 56 8 14336	AND absz,y 57 9 14592	NOP 58 : 14848	RLA absz,y 59 ; 15104	NOP-3 60 < 15360	AND absz,x 61 = 15616	ROL absz,x 62 > 15872	RLA absz,x 63 ? 16128
	PHA 72 H 18432	EOR # 73 I 18688	LSR A 74 J 18944	— 75 K 19200	JMP absz 76 L 19456	EOR absz 77 M 19712	LSR absz 78 N 19968	SRE absz. 79 O 20224
	CLI 88 X 22528	EOR absz,y 89 Y 22784	NOP 90 Z 23040	SRE absz,y 91 [23296	NOP-3 92 £ 23552	EOR absz,x 93] 23808	LSR absz,x 94 ↑ 24064	SRE absz,x 95 ← 24320
	PLA 104 26624	ADC # 105 26880	ROR A 106 27136	— 107 27392	JMP(absz) 108 27648	ADC absz 109 27904	ROR absz 110 28160	RRA absz 111 28416
	SEI 120 30720	ADC absz,y 121 30976	NOP 122 31232	RRA absz,y 123 31488	NOP-3 124 31744	ADC absz,x 125 32000	ROR absz,x 126 32256	RRA absz,x 127 32512
	DEY 136 f7 34816	NOP-2 137 f2 35072	TXA 138 f4 35328	— 139 f6 35584	STY absz 140 f8 35840	STA absz 141 SH-RET 36096	STX absz 142 NAGYBETŰ 36352	SAX absz 143 36608
	TYA 152 SZÜRKE 38912	STA absz,y 153 V. ZÖLD 39168	TXS 154 V. KÉK 39424	— 155 V. SZÜRKE 39680	— 156 BIBOR 39936	STA absz,x 157 CRSR← 40192	— 158 SÁRGA 40448	— 159 CIÁN 40704
	TAY 168 43008	LDA # 169 43264	TAX 170 43520	— 171 43776	LDY absz 172 44032	LDA absz 173 44288	LDX absz 174 44544	LAX absz 175 44800
	CLV 184 47104	LDA absz,y 185 47360	TSX 186 47616	— 187 47872	LDY absz,x 188 48128	LDA absz,x 189 48384	LDX absz,y 190 48640	LAX absz,y 191 48896
	INY 200 51200	CMP # 201 51456	DEX 202 51712	— 203 51968	CPY absz 204 52224	CMP absz 205 52480	DEC absz 206 52736	DCP absz 207 52992
	CLD 216 55296	CMP absz,y 217 55552	NOP 218 55808	DCP absz,y 219 56064	NOP-3 220 56320	CMP absz,x 221 56576	DEC absz,x 222 56832	DCP absz,x 223 57088
	INX 232 59392	SBC # 233 59648	NOP 234 59904	— 235 60160	CPX absz 236 60416	SBC absz 237 60672	INC absz 238 60928	ISC absz, 239 61184
	SED 248 63488	SBC absz,y 249 63744	NOP 250 64000	ISC absz,y 251 64256	NOP-3 252 64512	SBC absz,x 253 64768	INC absz,x 254 65024	ISC absz,x 255 65280

```
ORA abs z.
13          RETURN!
3328          CT M!
```

ASL és ORA

SRE — Shift Right and EOR Akku
LSR és EOR

SAX — Store Akku AND X
Logikai ÉS műveletet hajt végre az akku és az X regiszter között, az eredményt a címzésmód szerinti memóriacímre teszi

CRA — CRAsh
„Lefagyasztja" a processzor működését

NOP
NOP—2 Üres utasítás, a processzor erre az
NOP—3 utasításra érve nem hajt végre semmilyen műveletet, sőt az utasítástól függően még 1, illetve 2 bájtot nem vesz figyelembe

Az „új" utasítások eredményes alkalmazását egyelőre korlátozza az a tény, hogy pillanatnyilag nincs olyan assembler fordítóprogram, amely ismerné ezeket a kódokat. (Magyar szakemberek, figyelem! Itt a lehetőség egy ilyen program megírására! Egészen biztos, hogy lenne rá kereslet!) Az általam használt SMON monitorprogram is csak visszafordítani képes az új utasításokat, beírni nem lehet vele. Létezik ezenkívül egy FMON nevű monitorprogram, amely már ezt is tudja, azonban más elnevezéseket alkalmaz. Azoknak, akiket érdekel és esetleg a kezükbe kerül az FMON, felsorolom az általa használt mnemonikokat:

SMON	FMON
LAX	LTX
DCP	DEM
ISC	INB
RLA	RAN
RRA	RAD
SLO	ASO
SRE	ESE
SAX	AXX
CRA	ABS
NOP	NO1
NOP—2	NO2
NOP—3	NO3

A táblázatban használt rövidítések:
absz. = abszolút
zp. = zero page (Ø. lap)
CT v. CTRL = CONTROL
SH = SHIFT
v. = világos
s. = sötét

SZATMÁRI LÁSZLÓ

PROGRAMOK

COMMODORE 64

RESTORE n

Többször előfordult, hogy sok DATA közül egy adott csoportra volt szükségem. Ha cikluson belül többször kellett hozzáférni ugyanazokhoz az adatokhoz, akkor jó lett volna használni a RESTORE n utasítást, ezt azonban a C64 nem engedi meg. Ezért készítettem egy gépi kódú programot, amely helyettesíti a RESTORE n-et.

Három követelményt támasztottam a programmal szemben: könnyen kezelhető legyen; a HELP-pel végzett újrasorszámozás esetén az n értéke módosuljon; hibás használat esetén hibaüzeneteket adjon.

A program hívása:

SYS855:GOSUB n (n a kívánt sorszám)

Megszorítás, hogy a kívánt sornak DATA-val kell kezdődnie.

Az utasítás helytelen begépelésére a gép "?SYNTAX ERROR"-ral válaszol. Nem létező sorra hivatkozáskor "?UNDEF'D STATEMENT ERROR" jelenik meg. Amennyiben a hívott sor első utasítása nem DATA, akkor "?NINCS DATA A SORBAN" kerül a képernyőre.

Az *1. listán* az Assembly program, a *2. listán* pedig a bemutató program látható.

FEKETE IMRE

```
1000          100 ;
1000          110 ; DATA MUTATO ALLITASA
1000          120 ; A MEGHOZOTT SOR ELEJERE
1000          130 ;
1000          140 ; HIVASA:
1000          150 ;    SYS 855:GOSUB XXXXX
1000          160 ;
0357          170 *=$0357
0357 20 79 00 180      JSR $0079  ;SZINTAKTIKA VIZSGALATA
035A C9 3A    190      CMP #$3A
035C D0 56    200      BNE SYNT
035E 20 73 00 210      JSR $0073
0361 C9 83    220      CMP #$83
0363 D0 57    230      BNE SYNT
0365 20 73 00 240      JSR $0073
0368 20 8B A9 250      JSR $A98B  ;GOSUB UTANI SZAM ATALAKITASA BIN-
036B 20 F7 B7 260      JSR $B7F7  ;HISSA CIMKOLASA #0014-#0015 CIMEN
036E          270 ;
036E          280 ;AZ ADOTT SOR KERESESE
036E          290 ;
036E A5 2B    300 KEMES  LDA $2B
0370 85 FB    310      STA $FB
0372 A5 2C    320      LDA $2C
0374 85 FC    330      STA $FC
0376 A0 05 00 340 TESV   LDA #$05
0379 A9 00    350      LDY #$00
037B B1 FB    360      LDA ($FB),Y
037D C5 15    370      CMP $15
037F D0 25    380      BNE TOVABB
0381 A6       390      DEY
0382 B1 FB    400      LDA ($FB),Y
0384 C5 14    410      CMP $14
0386 D0 1E    420      BNE TOVABB
0388 C8       430      INY        ;DATA ALL-E A SOR ELEJEN?
0389 C8       440      INY
038A B1 FB    450      LDA ($FB),Y
038C C9 83    460      CMP #$83
038E D0 06    470      BNE UZENET
0390 B1 FB    480      LDA $FB
0392 18       490      CLC
0393 69 04    500      ADC #$04
0395 85 41    510      STA $41
0397 90 FC    520      LDA $FC
0399 69 00    530      ADC #$00
039B 85 42    540      STA $42
039D 85 14    550      STA $14
039F A6 15    560      LDX $15
03A1 85 2D    570      STA $2D
03A3 A0 01    580      STA $A6
03A5          590      RTS        ;KILEPES,HA MINDEN RENDBEN
03A6 A0 01    600 TOVABB LDY #$01   ;CIM BEALLITASA A KOV. SOR ELEJERE
03A8 B1 FB    610      LDA ($FB),Y
03AA AA       620      TAX
03AB          630      DEY
03AC B1 FB    640      LDA ($FB),Y
03AE 65 FB    650      STA $FB
03B0 86 FC    660      STX $FC
03B2 4C 76 03 670      JMP TESV
03B5 A0 01    680 HIBA   LDY #$01   ;VEGE VAN-E A PROGRAMNAK
03B7 B1 FB    690      LDA ($FB),Y
03B9 F0 06    700      BEQ BAJ
03BB 60       710      RTS
03BC          711 ;
03BC          712 ;HIBAUZENETEK
03BC          713 ;
03BC A2 00    720 SYNT   LDX #$00
03BE 4C 00 03 730      JMP $0300
03C1 A2 11    740 BAJ    LDX #$11
03C3 4C 00 03 750      JMP $0300
03C6 A9 D1    800 UZENET LDA #NEVC
03C8 85 22    810      STA $22
03CA A9 03    820      LDA #NEVC
03CC 85 23    830      STA $23
03CE 4C 47 B4 840      JMP $B447
03D1 4E 49 4E 850 NEV    .TEXT"NINCS DATA A SORBAN"
03E2 1E       860      .BYTE $1E
03E3          870      .END

LINES:97   SYMBOLS:0   ERRORS:0

BAJ  =03C1  HIBA =03B5  KEMES =036E  NEV  =03D1  SYNT =03BC
TOVABB=03A6  UZENET=03C6
```

1. lista

2. lista

```
10 OPEN4,4:PRINT#4," A";SPC(5);"B";SPC(5);"C"
20 PRINT#4,"-----------------"
30 DATA 1,2,3,4
40 DATA 5,6,7,8
50 FOR I=1 TO 5
60 SYS 855:GOSUB 40:READ A,B,C
70 PRINT#4,A;SPC(3);B;SPC(3);C
80 NEXT I
90 CLOSE 4
```

```
 A     B     C
-----------------
 5     6     7
 5     6     7
 5     6     7
 5     6     7
 5     6     7
```

Mit ismer a HELP + ?

Az 1986. márciusi számukban „Floppykezelés Kernal-rutinok segítségével" címmel közreadott cikkben téves információkat jelentettek meg. A HELP + nevű program assembler értelmező része igenis ismeri a cikkben egyébként hiányolt < > jeleket. Ezeknek felhasználásához mellékelem a következő assembler-listát, ami a nevezett program assemblerével készült.

ÁDÁMKÓ LAJOS
Ózd

```
PROBA3

C000              100          *=$C000
C333              110 ALAP     *=$C333
C000 A9 C3        120          LDA #ALAP>
C002 8D 00 C1     130          STA $C100
C005 A9 33        140          LDA #ALAP<
C007 8D 01 C1     150          STA $C101
C00A A2 C3        160          LDX #ALAP>
C00C A0 33        170          LDY #ALAP<
C00E 8E 02 C1     180          STX $C102
C011 8C 03 C1     190          STY $C103
C014 60           200          RTS
C015              65535        .END

ZEILEN:12   SYMBOLE:1   FEHLER:0

ALAP  =C333
```

Képernyőn a Primo agya!

A számítógép belső világának, tárolóterületeinek, adatmozgatásának áttekintése nem könnyű a kezdő számára. Ebben szeretnék segíteni programommal, amely a képernyőre vetíti a gép RAM-ját. Az pedig még a haladóknak is élmény lehet, hogy a kép él, mozog: megfigyelhető a rendszerváltozók és a verem állandó változása, az adatok, tömbök elhelyezkedése, mozgása. Gyakorlati célt nem szolgál a program, csak ismerkedés, barátkozás a géppel, bepillantás lelkivilágába!

A képernyőre vetítés érdekében a RAM-ot 6 kbájtra korlátoztam, így egyszerre jelenik meg a képernyő tetején a munkaterület (16384—17385), alatta beírt programunk szövege (17386-tól), a változó- és tömbtábla, a folyton izgő-mozgó verem, a sztringterület, a legalsó sorban pedig a rövid gépi kódú szubrutin, amely a képernyő-RAM-ba másolást végzi.

A gépi kódú szubrutin tömbmásolást végez:

```
LD HL, 4ØØØH   a RAM kezdete
LD DE          a képernyőtár kezdőcíme, a Primo
               típusától függően!
LD BC, 18ØØH   az átírandó terület mérete
LDIR
RET
```

A továbbiakat maga a program magyarázza meg. A kezdők tájékoztatására meg kell jegyezni, hogy a képernyőn a tárolók tartalma bitenként jelenik meg: a világító pont 1-et, a sötét Ø-át jelent.

FEKETE GYÖRGY

A szerkesztő megjegyzése: *Az érdekes és látványos, de a szerző szerint is gyakorlati célt nem szolgáló programnak van egy sokkal egyszerűbb, mégis többet tudó és gyakorlati hasznú DRAGON (TRS80 Color) változata. A program egyetlen utasításpárból áll, mellyel a memória (a teljes 64 kbájt!) tetszés szerinti 0,5 kbájtját jelölhetjük ki képernyőterületté, illetve ezt a kijelölést törölhetjük. A bekapcsolás alapcíme 65479, a kikapcsolásé 65478. Ezekhez kell hozzáadni*

$2 \cdot {}^{2}\log(4 \cdot M)$

ahol M a kijelzendő memória kezdőcíme kbájt egységekben. Az így kiadódó címre kell beírni egy Ø—255 közti értéket. Ennek a programocskának programok működés közbeni követésénél van szerepe.

```
10 REM ****************************
20 REM * KEPERNYON A PRIMO AGYA! *
30 REM ****************************
40 POKE 16561,127,87: CLEAR 1536
50 DEFINT A-F: DIM S(400), X$(100)
60 G=230+87*256: J=PEEK(20)
70 POKE G,33,0,64,17,0,J,1,0,24,237,176
80 POKE G+11,201: CLS: PRINT CHR$(6)
90 FOR D=0 TO 150: H=CALL(G): NEXT
100 PRINT@ 1,8,"    MUNKATERULET     "
110 PRINT@ 3,8," A PROGRAM SZOVEGE  ";
120 PRINT@ 6,8," VALTOZOK ES TOMBOK ";
130 PRINT@ 11,8,"     S T A C K      ";
140 PRINT@ 13,8,"    STRINGTERULET   ";
150 PRINT@ 15,8," GEPI KODU SZUBRUTIN ";
160 FOR D=0 TO 20000: NEXT
170 FORD=0 TO 150: H=CALL(G): NEXT
180 PRINT@ 6,8,"SZAMTOMBOT TOLTOK FEL";
190 FOR A=0 TO 6000: NEXT
200 FOR A=0 TO 400: S(A)=RND(9999)/7
210 H=CALL(G): NEXT
220 PRINT@ 7,8," STRINGTOMBOT TOLTOK ";
230 FOR B=0 TO 6000: NEXT
240 FOR C=0 TO 100
250 X$(C)=STRING$(12,RND(220)+35)
260 H=CALL(G): NEXT
270 PRINT@ 7,8,"ATRENDEZEM A SZAMOKAT";
280 FOR C=0 TO 6000: NEXT
290 PRINT@15,8,"LOMBNYOMASRA FIGYELJEN";
300 POKE G+8,22
310 FOR A=0 TO 199: S=S(A+200)
320 S(A+200)=S(A): S(A)=S: H=CALL(G)
330 BEEP 1,1: IF INKEY$="" THEN 350
340 NEXT: GOTO 310
350 CLS: PRINT@ 7,15,"V E G E"
360 POKE 16561,255,J-1: CLEAR 50
```

COMMODORE 64

Üzemmódok keverése

```
CØØØ 78          SEI
CØØ1 A9 24       LDA #$24
CØØ3 8D 14 Ø3    STA $Ø314
CØØ6 A9 CØ       LDA #$CØ
CØØ8 8D 15 Ø3    STA $Ø315
CØØB A9 ØØ       LDA #$ØØ
CØØD 8D 12 DØ    STA $DØ12
CØ1Ø AD 11 DØ    LDA $DØ11
CØ13 29 7F       AND #$7F
CØ15 8D 11 DØ    STA $DØ11
CØ18 A9 81       LDA #$81
CØ1A 8D 1A DØ    STA $DØ1A
CØ1D A9 7F       LDA #$7F
CØ1F 8D ØD DC    STA $DCØD
CØ22 58          CLI
CØ23 6Ø          RTS
CØ24 AD 19 DØ    LDA $DØ19
CØ27 8D 19 DØ    STA $DØ19
CØ2A AD 12 DØ    LDA $DØ12
CØ2D FØ 12       BEQ $CØ41
CØ2F A9 1B       LDA #$1B
CØ31 8D 11 DØ    STA $DØ11
CØ34 A9 15       LDA #$15
CØ36 8D 18 DØ    STA $DØ18
CØ39 A9 ØØ       LDA #$ØØ
CØ3B 8D 12 DØ    STA $DØ12
CØ3E 4C BC FE    JMP $FEBC
CØ41 A9 3B       LDA #$3B
CØ43 8D 11 DØ    STA $DØ11
CØ46 A9 28       LDA #$28
CØ48 8D 18 DØ    STA $DØ18
CØ4B A9 CA       LDA #$CA
CØ4D 8D 12 DØ    STA $DØ12
CØ5Ø 4C 31 EA    JMP $EA31
CØ53 FF          ???
```

Az 1986/1. számban megjelent, Megszakítás II. című cikkhez kapcsolódó program a raszter szerinti megszakítást használva, a grafikus és a karakteres üzemmód keverését teszi lehetővé. Az alsó hat karaktersorban normál üzemmódban dolgozhatunk, fölötte pedig a nagy felbontású grafikus üzemmódot használhatjuk. A grafikus kép a memóriában 8 k—16 k, a hozzá tartozó színmemória pedig 2 k—3 k között található. A program hívása: SYS 49152 utasítással történik.

GYURICZA PÉTER
II. gimn. tanuló
Szolnok

COMMODORE 64

ROM-rutint bemutató program

A program a Ø81ØH címtől Ø8CBH-ig tart. Így lehetőség van arra, hogy elé egy BASIC indítóprogramot írjunk. A gépi kódú főprogram Ø81AH-tól kezdődik, tehát SYS 2Ø74-re. A program egy pár BASIC- és KERNAL-rutint és használatukat mutatja be (képernyőtörlés, véletlenszám-generálás, egésszé alakítás, lebegőpontos szorzás stb.).

A kezdeti értékek beállítása után végtelen ciklusban dolgozik. Ebből esetleges elágaztatásra az F1 gomb hatására lép ki. Jelen esetben egy BASIC melegstartot hajt végre, ami természetesen átírható akármásra. Ez az utasítás a memória Ø83BH címen helyezkedik el.

Minden ciklusban két véletlenszám generálódik. Az egyik Ø—1 közé eső számot 2Ø-szal, a másikat 12-vel szorozza meg, és egész részüket elhelyezi a memória egy-egy bájtján. Ezután a képernyő közepétől jobbra le, jobbra fel, balra fel és balra le szimmetrikusan, a két véletlenszámnak megfelelően a képernyőre kiírja az A6H értéknek megfelelő jelet, persze mindig más színnel. Ez érdekes hatást vált ki a képernyőn.

Azt, hogy milyen nagy legyen a kép, befolyásolhatjuk azzal, hogy a memóriában elhelyezett két lebegőpontos állandót megváltoztatjuk, ügyelve arra, hogy a vízszintes nagyság ne haladja meg a 2Ø-at, a függőleges a 12-t.

A vízszintes konstans Ø8A5H-tól kezdődően 5 bájt, a függőleges Ø8AB-től helyezkedik el. Ha kisebb a kép, akkor gyorsabb a változás. A Ø857H címen helyezkedik el a kiírandó karakter kódja. Ha ezt megváltoztatjuk A6H-ról például 71H-ra, akkor pöttyöket fog kiírni négyzetháló helyett.

A programot mindenki kiegészítheti saját ízlése szerint. Érdekes hatásokhoz juthatunk a különböző változtatásokkal.

AKLI PÉTER

```
.:Ø81Ø 2Ø A2 BB 2Ø 2B BA 2Ø CC
.:Ø818 BC 6Ø 2Ø 3E Ø8 2Ø B1 Ø8
.:Ø82Ø 86 15 2Ø BE Ø8 86 14 2Ø
.:Ø828 97 EØ A5 62 2Ø 9B Ø8 2Ø
.:Ø83Ø 5D Ø8 A5 C5 C9 Ø4 FØ Ø3
.:Ø838 4C 1D Ø8 6C Ø2 AØ A9 ØØ
.:Ø84Ø 8D 2Ø DØ 8D 21 DØ 2Ø 44
.:Ø848 E5 A9 ØC 85 16 A9 14 85
.:Ø85Ø 17 6Ø 18 2Ø ØA E5 A9 A6
.:Ø858 2Ø D2 FF 18 6Ø 18 A5 17
.:Ø86Ø 65 15 A8 A5 16 65 14 AA
.:Ø868 2Ø 52 Ø8 18 A5 17 65 15
.:Ø87Ø A8 18 A5 16 E5 14 AA 2Ø
.:Ø878 52 Ø8 A5 17 E5 15 18 A8
.:Ø88Ø A5 16 E5 14 AA 2Ø 52 Ø8
.:Ø888 18 A5 17 E5 15 A8 18 A5
.:Ø89Ø 16 65 14 AA 2Ø 52 Ø8 6Ø
.:Ø898 29 ØF C9 ØØ FØ Ø3 8D 86
.:Ø8AØ Ø2 6Ø 4C 1D Ø8 85 AØ ØØ
.:Ø8A8 ØØ ØØ ØØ 84 CØ ØØ ØØ ØØ
.:Ø8BØ ØØ 2Ø 97 EØ 2Ø ØC BC A9
.:Ø8B8 A5 AØ Ø8 4C 1Ø Ø8 2Ø 97
.:Ø8CØ EØ 2Ø ØC BC A9 AB AØ Ø8
.:Ø8C8 4C 1Ø Ø8 ØØ ØØ ØØ ØØ ØØ
.o
```

Megszakításvezérelt BREAK-figyelés gé

```
 10 **********************
 20 * GEPI KODU BREAK    *
 30 *       INTERRUPT    *
 40 * START:       64973 *
 50 * RUTIN CIME: )FDFD *
 60 * RUTIN VEGE: 65249 *
 70 * VECTOR CIME:)FC00 *
 80 * RAMTOP:      64511 *
 90 **********************
100 ADR     EQU  )FDCD
110 RAMT    EQU  64511
120         ORG  ADR
130         ENT  ADR
140         LD   HL,CONT
150         LD   IX,)FC00
160         LD   B,129
170 PUT     LD   (IX),H
180         LD   (IX+1),L
190         INC  IX
200         INC  IX
210         DJNZ PUT
220 * CIMTABLAZAT KESZ
230         LD   HL,RAMT
240         LD   (23730),HL
250         LD   (HL),)3E
260         DEC  HL
270         POP  DE
280         LD   SP,HL
290         LD   BC,)1303
300         PUSH BC
310         LD   (23613),SP
320         LD   A,)FC
330         LD   I,A
340         IM   2
350         EX   DE,HL
360         JP   (HL)
370 CONT    PUSH IX
380         POP  IX
390         POP  IX
400         PUSH IX
410         DEC  SP
420         DEC  SP
430         PUSH AF
440         PUSH HL
450         PUSH DE
460         PUSH BC
470         LD   HL,(23672)
480         INC  HL
490         LD   (23672),HL
500         LD   A,H
510         OR   L
520         JR   NZ,KEY
530         INC  (IY+64)
540 KEY     PUSH IX
550         CALL )02BF
560         EI
570         POP  BC
580         LD   A,B
590         AND  )C0
600         JP   Z,END
610         LD   A,)FE
620         IN   A,()FE)
630         RRA
640         JP   C,END
650         LD   A,)7F
660         IN   A,()FE)
670         RRA
680         JP   C,END
690         LD   A,(COUNTA)
700         CP   50
710         JP   NC,OFLOW
720         INC  A
730         LD   (COUNTA),A
740         JP   ENDY
750 OFLOW   DI
760         LD   A,2
770         CALL )1601
780         LD   B,6
790         LD   IX,TNAME
800         LD   A,0
810         LD   (FUG),A
820 CYKL    LD   A,)16
830         RST  )10
840         LD   A,(FUG)
850         RST  )10
860         LD   A,19
870         RST  )10
880         LD   (STBC),BC
890         LD   A,(IX)
900         RST  )10
910         LD   A,(IX+1)
920         RST  )10
930         LD   A,58
940         RST  )10
950         INC  IX
960         INC  IX
970         POP  BC
980         PUSH BC
990         CALL CONVER
1000        LD   B,C
1010        CALL CONVER
1020        LD   A,59
1030        RST  )10
1040        LD   DE,SPACE
1050        LD   BC,8
1060        CALL )203C
1070        POP  BC
1080        CALL )2D2B
1090        CALL )2DE3
1100        CALL )16BF
1110        LD   A,(FUG)
1120        INC  A
1130        LD   (FUG),A
1140        LD   BC,(STBC)
1150        DJNZ CYKL
1160        EI
1170        JR   ERROR
1180 CONVER LD   A,B
1190        AND  )F0
1200        RRA
1210        RRA
1220        RRA
1230        RRA
1240        CALL LOOK
1250        LD   A,B
1260        AND  )0F
1270        CALL LOOK
1280        RET
1290 LOOK   ADD  A,)30
1300        CP   )3A
1310        JR   C,SMALL
1320        ADD  A,7
1330 SMALL  RST  )10
1340        RET
1350 STBC   DEFW 0
1360 SPACE  DEFM "      "
1370        DEFB )16
1380 FUG    DEFB 0,27
1390 TNAME  DEFM "BCDEHLAFIXPC"

1400 * REG. KIIRATAS VEGE
1410 ERROR  RST  )08
1420        DEFB )14
1430 END    XOR  A
1440        LD   (COUNTA),A
1450 ENDY   POP  BC
1460        POP  DE
1470        POP  HL
1480        POP  AF
1490        POP  IX
1500        RET
1510 COUNTA DEFB 0
1520        DEFB 0
```

A program tesztelési segítséget jelent a Spectrum-felhasználók számára: a felhasználói (RAM-ban futó) programok futási ideje alatt figyeli a CAPS—BREAK gombkombinációt, és ezeknek 1 másodpercnél hosszabb ideig történő lenyomása esetén megszakítja az éppen futó programot, kiíratja a regiszterek tartalmát, és a rendszer hibakezelő rutinján keresztül visszaadja a vezérlést a felhasználónak.

Megkötések a felhasználói programra

1. A maszkolható megszakításoknak engedélyezettnek kell lenniük (nem adható ki a DI utasítás).

2. A 2-es megszakítási üzemmód nem használható, mert ez foglalt a felügyelő rutin számára.

A megszakítás elfogadásakor a képernyő jobb felső sarkába íródnak ki a főregiszterpárok IY kivételével, és PC jelzéssel az utasításszámláló. A kiíratás decimális és hexadecimális alakban történik. A képernyő alsó részében megjelenik a

'BREAK INTO PROGRAM'

hibaüzenet, hasonlóan a BASIC-BREAK-hez.

Implementáció

A program csak 48 k változaton fut, nem kompatíbilis a 16 k-s gépre. A szűken vett hossza 276 bájt, de a 2-es megszakítási üzemmódhoz szükséges ugrási címtáblázat további fél k-t foglal. Ezt a program az első futtatáskor magának felépíti. A táblázat címe: #FCØØ. Szintén az első futtatáskor beállítódik az új RAMTOP érték: 64511. Betöltési és indítási cím: 64973.

Futtatás

A BASIC-betöltő beírása és futtatása után mentsük ki a programot, majd RANDOMIZE USR 64973. Ezután már él a BREAK-figyelés. Üres gépen, saját programok tesztelése előtt, célszerű egy LOAD ... CODE: RANDOMIZE USR 64973-mal indulni, így már nem kell félni a ciklusba eséstől. A rutin felügyelete a következő NEW utasításig tart, nem szükséges tehát minden használat után újraindítani.

Futtatási megkötések

1. A RAMTOP fölötti kotorászás rendszerösszeomláshoz vezet(het).

2. Minden NEW utasításnál újra kell inditani a rutint.

NAGY PÉTER

ódú programokhoz

A BASIC-betöltő

```
1000 DATA 33,253,253,221,33,0,25
2,6,129,221,116,0,221,117,1,221,
35,221,35,16,244,33,255,251,34,1
78,92,54,62,43,209,249,1,3,19,19
7,237,115,61,92
1005 DATA 4813
1010 DATA 62,252,237,71,237,94,2
35,233,221,229,221,225,221,225,2
21,229,59,59,245,229,213,197,42,
120,92,35,34,120,92,124,181,32,3
,253,52,64,221,229,205,191
1015 DATA 6305
1020 DATA 2,251,193,120,230,192,
202,213,254,62,254,219,254,31,21
8,213,254,62,127,219,254,31,218,
213,254,58,224,254,254,50,210,69
,254,60,50,224,254,195,217,254
1025 DATA 7167
1030 DATA 243,62,2,205,1,22,6,6,
221,33,199,254,62,0,50,197,254,6
2,22,215,58,197,254,215,62,19,21
5,237,67,189,254,221,126,0,215,2
21,126,1,215,62
1035 DATA 5070
1040 DATA 58,215,221,35,221,35,1
93,197,205,162,254,65,205,162,25
4,62,59,215,17,191,254,1,8,0,205
,60,32,193,205,43,45,205,227,45,
205,191,22,58,197,254
1045 DATA 5476
1050 DATA 60,50,197,254,237,75,1
89,254,16,183,251,24,49,120,230,
240,31,31,31,31,205,179,254,120,
230,15,205,179,254,201,198,48,25
4,58,56,2,198,7,215,201
1055 DATA 5632
1060 DATA 0,0,32,32,32,32,32,22,
0,27,66,67,68,69,72,76,65,70,73,
88,80,67,207,20,175,50,224,254,1
93,209,225,241,221,225,201,0,0,0
,0,0
1065 DATA 3515
1070 LET S=64973: FOR I=1 TO 7
1080 LET C=0: FOR K=1 TO 40: REA
D A: POKE S,A: LET C=C+A: LET S=
S+1: NEXT K
1090 READ A: IF C<>A THEN  PRINT
 "HIBAS ADATSOR:";(I-1)*10+1000:
 STOP
1100 NEXT I: PRINT "MINDEN O.K."
1110 REM  SAVE "BREAK?"CODE 6497
3,276
1120 REM *Nagy Peter*
```


A NOVOTRADE RT

az alábbi szoftverek bemutatóját tartja a Hotel Rege

„KÁRTYA" termében
(Bp. II. ker., Pálos u. 2.)
de. 10 órától,
melyre
minden érdeklődőt
szeretettel várunk!

IX. 1.

Főkönyv- és folyószámla-könyvelési rendszer bemutatása

IX. 2.

Készletgazdálkodási rendszer

IX. 3.

Bér- és munkaügyi rendszer
Személyzeti rendszer

IX. 4.

Alap-, illetve fejlesztői rendszerek
és lokális hálózatok
IBM kompatíbilis
gépekből kialakítva

IX. 5.

Állóeszköz-gazdálkodás

NOVOTRADE

C64 tulajdonosok figyelmébe!

EXPANDER—64

Elsősorban a könnyű gépkezelést, a hatékony programírást támogatja 21 új BASIC utasításával. Alkalmazásakor a funkciógomboknak előre definiált jelentése van, mely tetszőlegesen módosítható a felhasználó által. Így egyetlen gombnyomással aktivizálhatunk egy utasítást.

Az EXPANDER—64 bővítménnyel kiegészített C64 összes hibaüzenete magyar nyelvű!

— **CATAL parancs:** megjeleníti a diszk katagólusát a képernyőn, anélkül, hogy a tárban levő program törlődne.

— **FINC parancs:** a programban megkeresi egy adott karaktersorozat előfordulási helyét.

— **MAX, MIN függvény:** vektorváltozók maximumát, illetve minimumát adja vissza.

— **WPOKE utasítás:** kétbyte-os szám memóriába írására szolgál.

— **WPEEK függvény:** kétbyte-os szám olvasása.

— **AT utasítás:** kiíratás kurzorpozicionálással.

— **MSAVE parancs:** kezdő és végcímével adott memóriatartomány (vagy program) diszkre, kazettára másolására szolgál. Monitorprogram helyett használható.

— **TRACE parancs:** a futó program végrehajtódásának követésére szolgál. A TROFF paranccsal megszüntethető.

— **APPEND parancs:** a programhozzátöltés tárban levő programhoz. Olyan felhasználók számára előnyös elsősorban, akik előre megírt szubrutinokat, programrészeket szeretnének a programjukhoz szerkeszteni.

segítségével egy karaktersorozat bármely betűjelét módosíthatjuk, felesleges értékadás nélkül.

— **POS utasítás:** visszaadja egy tetszőleges karaktersorozat előfordulási pozícióját egy másik karakterláncban.

— **DUMP parancs:** begépeléskor kiírja az összes addig használt változó pillanatnyi értékét.

— **HCOPY utasítás:** a teljes képernyő tartalma a printerre íródik.

— **GCOPY utasítás:** EPSON printerre kinyomtatja a nagyfelbontású grafikus képernyő tartalmát 320 × 200 pont méretben. Nagyfelbontású grafikát támogató utasítások:

— **GIN, GOFF:** grafikus képernyő be- és kikapcsolás
— **GCLEAR:** grafikus képernyő törlése
— **GCOL:** grafikus képernyő színbeállítása
— **GPLOT:** koordinátájával meghatározott grafikus képpont kigyújtása

Forgalmazza az Alkotó Ifjúság Egyesülés Számítástechnikai Iroda Budapest V., Garibaldi u. 2. Levélcím: 1519 Budapest, Pf. 330. Telefon: 112-666 Telex: 22-7272.

TPA felhasználóknak ajánljuk!

MULTIFMS, TÖBBTERMINÁLOS FORMÁTUMKEZELŐ RENDSZER

A MULTIFMS rendszer célja olyan software-környezet biztosítása a TPA felhasználó számára, amelyben könnyen készíthet Basic—Plus 2, Fortran—IV, illetve Fortran—IV—Plus többfelhasználós adatgyűjtő, rögzítő, lekérdező programrendszer(eke)t.

A MULTIFMS HASZNÁLATÁNAK ELŐNYEI:

1. **Egyetlen program több terminált szolgál ki.**
   A DOS—RV/OS—RV/E/ operációs rendszer alatt egy program (task) annyi példányban fut (foglalja a helyet a memóriában), ahány terminálról használják. Mivel a rögzítés, lekérdezés általában egy nagy számú terminállal egyidőben történik, software-segítség szükséges ahhoz, hogy az ilyen típusú feladatok megoldhatók legyenek. A MULTIFMS rendszer max. 16 terminál kiszolgálását vállalja egyetlen felhasználói programpéldánnyal.
2. **Formátumvezérelt képernyőkezelés.**
   A MULTIFMS formátumokkal dolgozik. A formátum egy képernyőterv jellemzőit tartalmazza tömör formában, elkészítése interaktív formátumszerkesztővel történik. A bevitt elemi adatokon formai és tartalmi ellenőrzéseket végez el, és az ellenőrzés eredményétől függően vezérli a további adatbevitelt. A MULTIFMS rendszer a képernyőtervezési fázistól kezdve a kitöltés vezérlésén át az elemi ellenőrzésekig hatékony segítséget nyújt a felhasználónak, nagymértékben csökkentve az ilyen típusú rendszerek készítéséhez szükséges időt. Mivel a képernyőkezelést a rendszer végzi, a felhasználói rendszerek képernyőkezelése egységes lesz, így a különböző rögzítéseket végző operátorok betanítása is leegyszerűsödik.
3. **Könnyen kezelhető rendszer.**
   A terminál kezelőjének nem kell ismernie az operációs rendszer bejelentkezési eljárását. A rendszer indítása központilag (pl.: operátori konzolról) történik. A MULTIFMS kapcsolja (attach-eli) a számára kijelölt felhasználói terminálokat. Azokat a terminálokat, amelyek csak a MULTIFMS-sel fognak kommunikálni, slave (szolga) állapotba lehet tenni, ezek csak a MULTIFMS futása idején fognak működni. Különösen távoli terminálóknál működő lekérdező rendszereknél hasznos ez a tulajdonság. A nem slave terminálok dinamikusan megszüntethetik (újrateremthetik) a MULTIFMS-sel való kapcsolatot.
4. **Dinamikus memóriahely-foglalás.**
   A MULTIFMS rendszer vezérlő részének (MLTFMS) memóriaigénye 10-12 kszó között van. Ez az állandó helyigény. A vezérlő rész a felhasználói terminálról érkező input alapján indítja, illetve kapcsolja az adott inputtal foglalkozó felhasználói programot, és a terminál számára memóriaterületet (ún. dynamic regiont) foglal le, amelybe majd a terminálhoz kapcsolható adatok kerülnek. Ennek a területnek a mérete a felhasználói program szükségletének függvénye. Ha egyidőben több terminálról érkezik azonos típusú (ugyanannak a felhasználónak szóló) input, azok feldolgozása változatlanul egyetlen programpéldánnyal történik, csak valamennyi terminál számára létrejön egy-egy region.
5. **Memória- és CPU-időtakarékos rendszerkialakítás.**
   Az elindított, de éppen használaton kívül levő felhasználói programok nem terhelik a CPU-t, szükség esetén checkpointolhatóak, a memóriába csak akkor kerülnek újra vissza, ha ténylegesen szükség van rájuk. Maga a vezérlő program is többnyire checkpointolható, azaz tologatható a memóriában.
6. **Rugalmas felhasználói rendszer kialakítási (bővítési) lehetőség.**
   Egy felhasználói rendszer a feladat természetéhez illeszkedő struktúrában egy vagy több programmal valósítható meg. A rendszernek lehetnek egymástól független (párhuzamos) programágai, valamint hierarchikus részei is.
7. **Több többterminálos rendszer párhuzamos futtatásának lehetősége.**
   A MULTIFMS-t több különböző program egyidejűleg használhatja, azaz több különböző rögzítési, lekérdező feladat végezhető el a gépen egyidejűleg.
8. **Funkciók megvalósítása szubrutinhívásokkal.**
   A formátumkönyvtárban tárolt formátumok és egy szubrutingyűjtemény segítségével a felhasználói programok a képernyőkezeléssel kapcsolatos tevékenységeket egyszerűen szubrutinhívásokkal valósítják meg.
9. **Könnyű és gyors formátumtervezés.**
   A formátum elkészítéséhez nincs szükség előzetes képernyőtervre, sem pedig speciális formátumleíró nyelvre. A formátum elkészítése interaktív formátumszerkesztővel történik (FED).
10. **Formátumfüggetlen programkészítés.**
    Az elkészített formátumhoz rendelt értékek (sorszám, oszlopszám, kiírandó szöveg, kezdőérték, igazítás, feltöltés stb.) nem függenek a felhasználói programtól, így azok módosításakor nem kell a programot megváltoztatni (újraépíteni).

A MULTIFMS csak lehetőséget teremt hatékony többterminálos rendszerek felépítéséhez. A tényleges hatékonyság nagymértékben függ a felhasználói programok megírási módjától, a funkciók megfelelő módon való széttagolásától, a felhasználói rendszer file-jainak szerkezetétől és a hozzáférési módtól.

Forgalmazza az
Alkotó Ifjúság Egyesülés
Számítástechnikai Iroda
Budapest V., Garibaldi u. 2.
Levélcím: 1519 Budapest, Pf. 330.
Telefon: 112-666
Telex: 22-7272

PTA–4000

Csatlakozók

A rendszer ismertetését a gép belsejében található két csatlakozósáv vonalainak tárgyalásával fejezzük be.

A 40 vonalas csatlakozó vezetékeit az *1. táblázat* tartalmazza. A jelölések:
1 — tápfeszültség
2—3 — kiválasztó
4—5 — címkijelölő (0000H—3FFFH ill. 6000H—6700H)
6 — kiválasztó a WAIT-jelhez
7—14 — adatbusz
15 — letiltás
16—19 — címkijelölő (4800H—4FFFH, 5000H—57FFH, 5800H—5FFFH, 8000H—BFFFH)
20—21 — tápfeszültség
22—37 — címbusz
38 — kimenetletiltás
39 — olvasás/írás
40 — tápfeszültség

A 60 vonalas csatlakozót a *2. táblázat* összegzi. A jelölések:
1—8 — címbusz
9—10 — üres
11—12 — tápfeszültség
13—14 — üres
15—16 — kiválasztó
17—24 — adatbusz
25 — letiltás
26 — külső WAIT-jel
27 — kazettabemenet
28 — WAIT-állapot bemenet
29 — kazettakimenet
30 — megszakításkérés
31—38 — címbusz
39—40 — üres
41—48 — tápfeszültség
49 — üres
50 — tápfeszültség-kimenet
51 — órajel
52—55 — tápfeszültség
56 — mint 6
57 — olvasás/írás
58 — mint 6, csak az ME1-lapra
59 — az ME1-lap kiválasztó
60 — mint 38

1. táblázat

1	VCC
2	PV
3	PU
4	Y0
5	S4
6	DME0
7—14	D7—D0
15	INHIBIT
16—18	S1—S3
19	Y2
20	VGG
21	GND
22—37	AD15—AD0
38	OD
39	R/W
40	GND

2. táblázat

1—8	AD7—AD0
9	PB0
10	PC7
11—12	VCC
13—14	NC
15	PV
16	PU
17—24	D7—D0
25	$\overline{\text{INHIBIT}}$
26	$\overline{\text{WEX}}$
27	$\overline{\text{CMTIN}}$
28	W1
29	CMTOUT
30	INT
31—38	AD8—AD15
39	PB1
40	NC
41—42	VCC
43	F.GND
44—48	VBAT
49	NC
50	BFO
51	ΦOS
52—55	GND
56	DME0
57	R/W
58	DME1
59	ME1
60	OD

Szoftver

A szoftver ismertetését a rendszerveremben tárolt állandók bemutatásával kezdjük. Ezek a 786BH—7B0CH cimtartományban találhatók. A címeket és a tárolt állandók jelölését a *3. táblázat* tartalmazza. Az egyes jelek magyarázata:
— külső/„beep", ki/be kapcsoló,
— minden kétbájtos tárolóban az első a magasabb helyiértékű.

A következő részben a BASIC sajátosságaival foglalkozunk. A gépi kódú programozáshoz szükséges segédlet, a mikroprocesszor kódtáblázata a HCC-nél önköltségi áron kapható.

3. táblázat

	786BH—	RMT/BEEP
	7871H—	WAIT Y/N (WAIT—0, WAIT0—3, WAIT1—2)
7872H—	7873H—	WAIT COUNTER
	7874H—	CURSOR ENABLE (1)
	7875H—	CURSOR POINTER (0—155)
	787DH—	BLINK CHARACTER
787EH—	787FH—	BLINK CURSOR
	788DH—	TRACE
	788EH—	TRACE CONDITION
	788FH—	OUTPUT BUFFER POINTER
	7890H—	FOR POINTER
	7891H—	GOSUB POINTER
	7894H—	STRING BUFFER POINTER
	7895H—	USING F/F
	7896H—	USING M
	7897H—	USING §
	7898H—	USING m
7899H—	789AH—	VARIABLE POINTER
	789BH—	ERL
789CH—	789DH—	CURRENT LINE
789EH—	789FH—	CURRENT TOP
78A0H—	78A1H—	PREVIOUS ADDRESS
78A2H—	78A3H—	PREVIOUS LINE
78A4H—	78A5H—	PREVIOUS TOP
78A6H—	78A7H—	SEARCH ADDRESS
78A8H—	78A9H—	SEARCH LINE
78AAH—	78ABH—	SEARCH TOP
78ACH—	78ADH—	BREAK ADDRESS
78AEH—	78AFH—	BREAK LINE
78B0H—	78B1H—	BREAK TOP
78B2H—	78B3H—	ERROR ADDRESS
78B4H—	78B5H—	ERROR LINE
78B6H—	78B7H—	ERROR TOP
78B8H—	78B9H—	ON ERROR ADDRESS
78BAH—	78BBH—	ON ERROR LINE
78BCH—	78BDH—	ON ERROR TOP
78BEH—	78BFH—	DATA POINTER
	79D1H—	OPN DV
79E0H—	79E1H—	USER COUNTER X
79E2H—	79E3H—	USER COUNTER Y
79E4H—	79E5H—	SCISSORING COUNTER Y
	79E6H—	ABSOLUTE POSITION X
79E7H—	79E8H—	SCISSORING COUNTER X
	79EAH—	LINE TYPE
	79EBH—	DOT LINE COUNTER
	79ECH—	UP/DOWN
	79EDH—	X MOTOR HOLD COUNTER
	79EEH—	PORT C
	79EFH—	Y MOTOR HOLD COUNTER
	79F0H—	GRAPH/TEXT
	79F2H—	ROTATE
	79F3H—	COLOR
	79F4H—	CSIZE
	79FFH—	LOCK
7B00H—	7B07H—	RND NUMBER
7B0AH—	7B0CH—	AUTO P—OFF COUNTER

PTA—4000 Fedélzeti számítógép az autóban

A jövő század számára elképzelt szériakocsikban, de már az autószalonok legújabb csodáiban is jelentős szerepet kapnak a szilíciumlapkák, a chipek felhasználásával kialakított különböző célú elektronikus készülékek, amelyek közül néhány már valóban megérdemli a fedélzeti számítógép elnevezést. Ha a fantáziát nem korlátozzák gyakorlati és gazdasági megfontolások, akkor a gépkocsiba minden beépíthető, és az autó alkalmas arra is, hogy a kívánalmaknak megfelelő gurulő elektronikus laboratóriumot alakítsunk ki benne. Az azonban nyilvánvaló, hogy a gépkocsi nem mindenben tekinthető optimális helynek a vezetést és üzemeltenőrzést segítő vagy egyéb elektronikus készülékek széles körű alkalmazására. Erre elég példaként felhozni, hogy a ma már több évtizedes alkalmazási tapasztalatok ellenére sem általánosak a magyar utakon futó szériakocsikban az elektronikus gyújtás és az érintkezőmentes töltésszabályozás készülékei, csupán ezeket említve a már megoldott eljárások közül.

A gépkocsiban alkalmazható elektronikus berendezéseket két csoportra oszthatjuk. Az egyik csoport készülékei közvetlenül a motorhoz vagy a fékrendszerhez kapcsolódva irányítástechnikai feladatokat oldanak meg. Ide tartoznak a blokkolásmentes fékrendszerek elektronikái, a benzinbefecskendezés szabályozói, a gyújtásszabályozó és -megszakító rendszerek. A másik csoport készülékei az ellenőrző funkciót látják el, és kiegészítő információkat szolgáltatnak a vezető vagy üzemeltető számára. Most ennek az utóbbi csoportnak a készülékeivel foglalkozunk, hiszen jóval többen lehetnek olvasóink között olyanok, akik meglévő gépkocsijukat kívánják jobb hatásfokkal üzemeltetni, és elérhetik, hogy személyiszámítógép-beszerzésüket ne a család színes videojáték-igénye, hanem a hálózati feszültségtől független és gépkocsiban történő alkalmazhatóság befolyásolja.

Gépkocsikba a műszerezés kiegészítésére különböző funkciójú, inkább díszítésre alkalmas készülékek vásárolhatók, amelyekre általában jellemző „az erdőben elvesztett tűt a tábortűznél kereső kiscserkész esete": nem a szükséges, hanem az egyszerűen mérhető paraméterek jelzését biztosítják. Van, aki a fordulatszámmérőre esküszik, mások a szívótorokban mérhető nyomást favorizálják. A témában elmélyedők azonban igen hamar tapasztalják, hogy a pillanatnyi üzemanyag-fogyasztást mérő érzékelő nélkül nem sokra lehet menni, még számítógépes feldolgozással sem.

Cikkünknek az ad aktualitást, hogy hazai gyártásból megfelelő számítógép és az NDK-ban gyártott, az újabb Wartburg gépkocsikba már be is szerelt üzemanyagszenzor rendelkezésre áll.

Miért előnyös?

A PTA—4000 számítógépből kialakított, a Wartburg szenzort tartalmazó rendszer leírása előtt tekintsük át azokat a feladatokat, amelyek többletszolgáltatásként elvárhatók egy viszonylag olcsó szériakocsiba utólag beépíthető, fedélzeti számítógéptől:

— mutassa az indulástól megtett utat km-ben, legalább 10 méteres felbontásban,

— mérje és jelezze az út során elfogyasztott üzemanyag mennyiségét,

— tegye meg ugyanezt a tankolási időponttól, illetve egy kiválasztott kezdőponttól számítva,

— adjon információt egy adott indulási ponttól az átlagsebességre vonatkozóan,

— jelezze a pillanatnyi futásra jellemző fogyasztást 100 km megtett útra vonatkoztatva,

— jelezze a tankban még rendelkezésre álló üzemanyag mennyiségét,

— jelezze, hogy ezzel az üzemanyagmennyiséggel, alapul véve az utolsó 100 km megtett út fogyasztását, még milyen távolságot futhatunk be,

— rendelkezzen az átlagos vezető látási viszonyaihoz alkalmazkodó digitális és analóg kijelzési lehetőséggel,

— szolgáltassa a pontos időt,

— adjon lehetőséget arra, hogy a vezető kívánsága szerint, értékelhető formában kijelezze a kívánt funkciók végeredményét.

A sort még folytathatnánk különböző igényekkel, de mutatóba ennyi is elég. Teljessé tehetjük a felsorolást azzal, hogy megkívánjuk a számítógéptől, hogy a fel nem sorolt újabb funkciókat szoftverúton tegye lehetővé. Ilyen lehet az analóg kijelzés, ahol egy megengedett érték túllépése módosítva jelenik meg, az előre programozott üzemanyag-fogyasztás vagy út-idő diagramtól való eltérés jelzése stb.

Mi kell hozzá?

Az igényeink teljesítéséhez minimálisan három bemenőjel szükséges az időalap, az üzemanyagszenzor és a pillanatnyi üzemanyag-fogyasztás mérésére, továbbá egy úttávadó. Számítógépként olyan eszköz jöhet szóba, amelynek fogyasztása a gépkocsi álló állapotában az akkumulátor önkisülésének nagyságrendjébe esik, és az igényeknek megfelelő kijelzést produkálja, valamint elhelyezhető a gépkocsiban, kellő rálátást adva a vezető és esetleg az utas számára is.

A jámbor óhajon, külföldi napidíjak számlálgatásán és a szakirodalom tanulmányozásán felül néhány éve véletlen szerencse folytán a Dobi István úti autópiacon találkoztam egy eladásra kínált originál cso-

magolású autó-számítógéppel, amelyhez üzemanyagmérő is tartozott. Tulajdonosa, aki valami egyszerű eszközt várt ajándékba a kellően felcicomázott kocsijába, igyekezett túladni rajta, miután túlságosan bonyolultnak találta a készüléket és a beszerelési és üzemeltetési utasítást is, ami természetesen csupán angol nyelven állt rendelkezésre. Így a készüléket igen jutányos feltételekkel (1 Ft = 1,2 osztrák schilling) sikerült megvásárolnom. Részletes szétszedéssel együtt járó hardvervizsgálat után, amit szerencsésen túlélt, üzembe is állítottam a számítógépet. Ez a Lada 1200 típusú gépkocsimba épített készülék a felsorolt igények zömét részben teljesíti. Már több mint két éve használom, és eltekintve a szerviz utáni újrabeállítgatások kényelmetlenségeitől, ami az akkuszerviznél történő teljes lekapcsolása miatt szükséges, a készülék hibamentesen működik, és a szolgáltatott adatok alapján bármikor eldönthetem hosszabb vidéki utakon, hogy hány forintomba kerül, ha 10 perccel hamarabb érkezem meg. Ennél lényegesebb azonban az, hogy kiválasztott útszakaszok fogyasztásait figyelve elég pontos információt kapok a motor és a karburátor megfelelő beállításáról, az utánállítás szükségességéről.

A készülék három egységet tartalmaz. A bolygógolyós üzemanyagszenzor az AC-pumpa fölé beépíthető, és az átfolyt üzemanyag-mennyiséggel arányos számú impulzust ad. Az úttávadó az alvázhoz erősített vasmagos tekercs, melynek mágneses terét a kardánra ragasztott mágnesek befolyásolják. A harmadik egység egy 5 dekádos plazmakijelzőt és 4 bites NS-processzort tartalmazó mikrogép.

Az üzemanyagszenzor

Érzékelője egy körpályára kényszerített golyó, amelyet az érintőlegesen bevezetett üzemanyag egy része magával ragad és keringtet. Az üzemanyag kivétele a bevezetéssel szinte azonos ponton, a pályára merőlegesen történik. A fekete műanyag golyó egy, a szenzorba épített infrakaput zár, aminek érzékelőjéről impulzusokat kapunk. A készülék a leírás szerint gyárilag nincs hitelesítve, és az előírás az, hogy a tankolásnál betöltött mennyiségekkel kell a kalibrációját korrigálni, amennyiben arra szükség van. A 30 literes tankolásoknál általában 1 literes eltéréseket tapasztaltam a készülék által mutatott és a benzinkútnál betöltött értékek között. Ez a töltőállomásoknál megkívánt pontossági osztályhoz közel álló eredmény. A szerkezet kialakítása olyan, hogy a benzin útját semmilyen formában sem tudja elzárni, hiszen a golyó átmérője kisebb, mint a körpályáé.

A megtett utat mérő szenzor a kardántengelyre erősített 4 mágnesből és az alvázhoz csavarozott tekercsből áll. Itt is szoftverúton történik a kalibráció, és azt 2 vagy több kilométer lefutott távolság alapján lehet elvégezni.

Az akkura állandóan rákötött tápvezetéken és a szenzorok bemenővezetékein felül még a gyújtás- és világításkapcsolóról van csatlakozás, amely arra szolgál, hogy a készülék plazmakijelzőjének csak üzem közben adjon tápenergiát, és fényerejét csökkentse az esti vagy éjszakai vezetésnél.

A kijelző

Öt decimális helyértékből áll, tizedesponttal ellátva, ami csak értékeket közöl, így emlékezni kell arra, mi volt a készülékhez intézett utolsó kérdés. A kérdések az előlapon rendelkezésre álló nyomógombokkal aktiválhatók. A beállításhoz takart nyomógombok állnak rendelkezésre. A kezelési utasítás többször figyelmeztet arra, hogy beállítani csak álló gépjárműben szabad, hiszen vezetni és programozni egy időben veszélyes művelet, bár itt a programozás csak a távolságérzékelő és benzinmérő szenzorának állandóit, valamint az idő beállítását jelenti.

A készülék az eredetileg programozott funkciókon felül továbbiak elvégzésére nem alkalmas. Ezek az 1—6 és 9 pontnak felelnek meg. A kijelző napsütésben nagyon nehezen olvasható, és a pillanatnyi fogyasztásnál a készülék nem veszi figyelembe a gyorsulásra fordított energiát. Erre akkor van szükség, ha a motor hatásfokára kívánunk következtetni.

A készülék szenzorai, amelyeket a Ladába beépítettem, már lehetőséget adtak arra, hogy egy kis fogyasztású számítógéppel a kívánságaimnak megfelelő rendszert alakítsunk ki. A rendelkezésünkre álló több lehetséges alternatíva közül (HP—41cx IL csatolóval, CASIO PB7Ø0, CASIO FX—8Ø2P, HD—75, HP—71B + IL, SHARP—PC15ØØA) többet részletesen vizsgáltunk. A leggyorsabb eredményt a HP-gépek alkalmazása igérte, mivel rendelkezésre álltak a HP—IL csatolóhurokhoz illesztett számlálók. A szenzorokról kapott impulzusokat ICM7227 típusú számlálókkal megszámolva, a számlálókat HP—82166 modullal az IL hurokhoz csatolva kialakítottunk egy olyan elrendezést, amely a feladatnak megfelelően működött, és igen egyszerű, BASIC-ben elvégezhető programozási lehetőséget adott.

A kísérlethez a forgószárnyas vízsebességmérők hitelesítésére kialakított berendezést alkalmaztuk néhány napig. A HP-gépekhez sajnos nem sikerült megszerezni a gépi nyelvű programozáshoz szükséges eszközöket, így a BASIC-ben megírt programok az eredeti készüléknél valamivel lassabban működtek, de lehetőséget adtak a HP—82161 típusú kazettás tároló alkalmazására, amire kb. 100 kbájt rögzíthető. Miután egy ilyen elrendezés ára magasabb, mint a gépkocsié, gyakorlati felhasználása, eltekintve néhány tesztelési feladattól, illuzorikus.

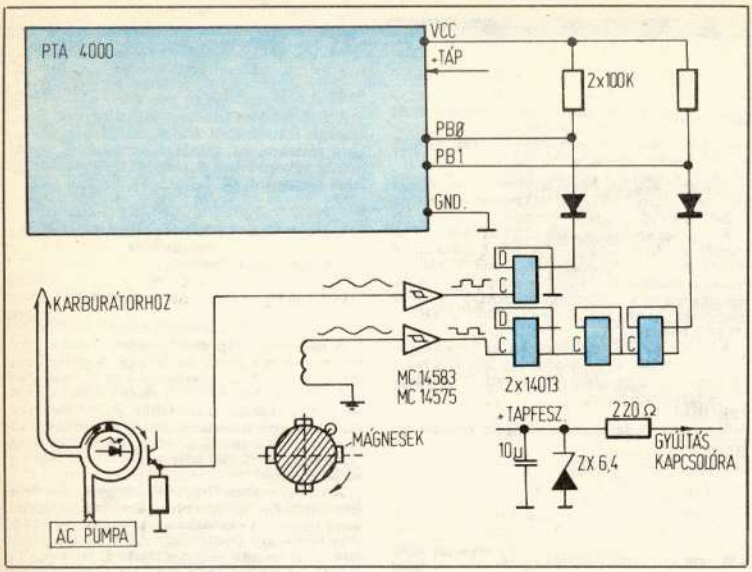
Hazai géppel

Ezután a SHARP—PC1500A géppel kezdtünk hozzá az ismertetett berendezés szolgáltatásainak bővítéséhez és a felsorolt specifikációnak teljesen megfelelő rendszer létrehozásához. A munka kezdetén még nem tudtuk, hogy a PC1500A-nak megfelelő, de annál nagyobb memóriakapacitású gép, a PTA—4000 + 16k hazai gyártásból is rendelkezésre fog állni, és így ez a munka az egyéni autószámítógép-hobbin túlmenő jelentőséget is kaphat. A SHARP-gép választásában igen jelentős volt az a tényező, hogy sikerült hozzájutni a gép belső dokumentációjához, ami lehetőséget adott arra, hogy ne az IL csatolón keresztül, hanem közvetlenül kapcsoljunk két hardverszámlálót egy HP-géphez. Az első néhány összeállításnál itt is az IL hurkon keresztül vettük igénybe a gépet.

Csak ezután következett a szoftverszámláló kialakítása. Az üzemanyagszenzor jele a PBØ, az útadó jele a PB1 csatlakozópontra került. A védelem és az egyszerűbb programozhatóság érdekében a jeleket egy 4Ø13 típusú CMOS FF áramkörrel leosztottuk. Így már lehetőség van arra is, hogy felhasználjuk a gép kerneljének rutinjait, ami igen előnyös egy kísérleti összeállításban.

A PTA—4000 gépkocsiban történő próbája közben szükségessé vált, hogy a gépet ne csak fedélzeti számítógépként, hanem eredeti funkciójának megfelelően, zsebszámítógépként is felhasználhassuk. Ehhez azonban további dekódert, ROM és esetleg RAM elemet tartalmazó hardver kell. A gép rendszerének ismeretében a feladat bizonyos korlátokkal szoftverúton is megoldható, ha a műszerállandókat a gép által nem használt, pontosabban a második karakterkészlet számára fenntartott memóriaterületen helyezzük el. Ez a megoldás azonban korlátozottnak tűnt, így inkább 2 k EPROM és RAM alkalmazása mellett döntöttünk, amely a H8ØØØ címre helyezhető el. Ha a gép fő funkciójának a gépkocsiban történő üzemelést tekintjük, akkor erre a hardverre természetesen nincs szükség.

Az alkalmazott gépi kódú program folyamatosan visszatér a ME1 HFØØF címen jelentkező bemenőpontok olvasására, és impulzust jelez, ha eltérnek a beolvasott jelek az előző ciklusban beolvasottól. Ezután arra van szükség, hogy a programot a valós idejű operációnak megfelelően építsük fel úgy, hogy a mintavételek közé beférjenek a kívánt funkciók. A szoftvermunkát igen egyszerűsítette volna a lapban már ismertetett MC 14534 típusú számlálók alkalmazása az IFSP illesztő felhasználásával, de ér-



A PTA—4000-es print out listája. A dátum kiírása a TIME funkció szerinti: 5. hó, 26. nap, 12 óra, 3 perc, 8 mperc.

demben vonzónak tűnt szoftverszámlálók kialakítása a feladat megoldásához. A számláló megkívánt mintavételi sebessége 6000/perc = 100/s maximális motorfordulatnál a kardánra helyezett 4 mágnessel és 2-es frekvenciaosztással számolva kisebb kell hogy legyen, mint 5 ms, ami csak igen körültekintő rendszerszervezéssel teljesíthető. Segít a helyzeten egy további 14Ø13 osztóként történő alkalmazása (lásd az *ábrát*), ami a rendelkezésre álló időt 20 ms-ra emeli. Felmerült annak a lehetősége is, hogy a maszkolható megszakítást használjuk fel a számlálásra, amit az üzemanyag és távolságszenzor jelével aktiválunk. A megoldást azért vetettük el, mert ez a pont nincs kivezetve a csatlakozóra, így nem használható a gépbe történő belenyúlás nélkül. A BRK/ON gombbal aktivált megszakítást egyébként felhasználtuk a programban. Használatáról annyit kell tudni, hogy a megszakítás akkor megy a felhasználó által a H79DB,DC memóriahelyen megadott címre, ha előzőleg H55-öt írunk a H79DA-ra. Ezt meg kell ismételni a bekapcsolás után, mert a tápfeszültség megjelenése után végrehajtott RESET rutin törli a H79DA memóriahely tartalmát.

Itt kell megjegyezni, hogy a PTA—4000 POKE utasítása megfelelő háttér-dokumentációk nélkül kellemetlen meglepetések forrása lehet, amit még az ALL RESET sem szüntet meg minden esetben. Annak érdekében, hogy a fejlesztés során fellépő elkerülhetetlen végtelen hurkokból újra feltámasztott gépünkkel azonos helyzetből induljunk újra, egy ÚTHENGER-nek elnevezett gépi programot is készítettünk, amely néhány bájt kivételével nullára írja a teljes tárterületet és kikapcsolja a gépet. Az új bekapcsolás a program lefutása után az ALL RESET-nek megfelelően jelentkezik be. A maszkolt megszakítás felhasználói rutinra történő elágaztatásával különböző rutinokat alakíthatunk ki, amelyekkel megakadályozhatjuk, hogy nagyobb programjainkat megszakításokkal befolyásolják, erről azonban helyhiány miatt most nem tudunk részletes információt adni.

Felhasználói programok

A PTA—4000-hez készített felhasználói-program-variációink mind számban, mind effektív bájthosszban egyre inkább duzzadtak, és kezelésük egyre lassúbbá vált a magnetofonkazetta adathordozóként történő alkalmazása miatt. A gyorsabb munka érdekében az IFSP használatával egy RS232 interfészen keresztül a PTA—4000-et C64 rendszerrel működtettük együtt, ami némileg megkönnyítette a már elkészült gépi rutinok visszakeresését az 1541-es floppyról. Ekkor adódott az az ötlet, hogy ki kellene hagyni a C64-et, az IFSP illesztőt és természetesen a C64 RS232 csatolóját is, megtanítva a PTA—4000-et a C64 soros buszának és így a C64-hez kapcsolt perifériáknak a kezelésére. A munka sikerrel járt, és a KA 160 illesztőt úgy módosítottuk, hogy a kazettás magnetofon REMOTE csatlakozóit felhasználva, a C64-hez a soros IEC buszon csatlakozó perifériákat kezelni tudjuk.

Miután sikerült, megírtuk és teszteltük a legkritikusabb, a működést bizonyító programrészeket. A Híradástechnika Szövetkezet rendelkezésünkre bocsátott egy PTA—4000+16 gépet és egy KA 160 típusú nyomtatót, amit egy Wartburg taxiba építettünk be. A Wartburgban üzemanyagszenzor rendelkezésre áll, és a leplombált taxióra hitelesen bizonyíthatja a berendezés üzembiztonságát. A taxiban elhelyezett gép nyomtatóval együtt üzemel, és a szenzorok csatlakoztatására a floppyillesztéshez kialakított soros busz DATA és CLOCK vonalait alkalmazzuk. Ennek megfelelően természetesen módosítanunk kell a már elkészített programjainkat, de addig is értékes információkkal gazdagodunk arra nézve, hogy a KA 160 nyomtató és a PTA—4000 hogyan viselkedik egy vibrációnak erősen kitett környezetben.

A PTA—4000+16 géppel megoldhatók a menetdiagram-regisztrálások, és az adatok floppyra vagy akár kazettára olvasása lehetőséget ad központi ellenőrző rendszerek kialakítására is, az eredetileg általunk a HP-gépekkel megvalósított műszaki paraméterek mellett. Ehhez azonban még igen jelentős szoftvermunkát kell elvégeznünk.

A hobbiból megkezdett feladatunk elvégzésében jelentősen túlmentünk azon az eredeti célkitűzésen, hogy napfényben is le tudjuk olvasni a ZONIC plazmakijelzőjét, és egy gyorsulással kompenzált fogyasztási rutint alkalmazzunk a már beépített szenzorokhoz.

PÁLOS LÁSZLÓ—ARDAY ZSOLT

...egy könyvet

mely több, mint négyszáz oldalon, nyolc nagyobb, jól megírt, jól strukturált és jól olvasható, kipróbált programot is tartalmaz.

Erről ismét eszünkbe jutott, hogy nem lehet nyelveket tanulni jó olvasókönyvek nélkül. Ez igaz a számítástechnikára, a programozási nyelvekre is. Miller könyve jó olvasókönyv. Nem követi például az e lap hasábjain is meghonosodott rossz hagyományt, hogy a kurzort mozgató karaktereket rosszul olvasható grafikus jelekkel listázza. Ehelyett minden programját valahogy így kezdi:

```
10Ø rem *** — a program neve — ***
11Ø:
12Ø:
13Ø rem **— inicializálás — **
14Ø home$=chr$ (147) : rem clr/home
                       képernyőtörlés
15Ø::cl$=chr$ (17) : rem kurzor le
16Ø::cf$=chr$ (145) : rem kurzor fel
17Ø::cb$=chr$ (157) : rem kurzor balra
18Ø::cj$=chr$ (29) : rem kurzor
                     jobbra
19Ø::fe$=chr$ (144) : rem fekete
  .
  .
  .
24Ø poke 53280, 14 : rem a képernyő
                     széle világoskék
  .
  .
25Ø poke 53272, 23 : rem kis- és
                     nagybetűs írásmód
  .
  .
  .
stb.
```

Ennek megfelelően a képernyőt például így kezeli:

```
  .
  .
  .
17ØØØ rem ** — az alprogram neve — **
17Ø1Ø print home$ : rem a képernyőt
                    töröltem, kurzor a
                    bal felső sarokban
17Ø2Ø cl=2 : rem 2 sorral lefele,
             paraméter a szubrutinnak
17Ø3Ø gosub 9ØØØ : rem ez mozgatja cl
                   számú sorral lefele
                   a kurzort
17Ø4Ø print "Ide jön a megjeleníteni
            kívánt szöveg"
  .
  .
  .
  .
  .
9ØØØ rem ** — a kurzort lefelé mozgató
             szubrutin — **
9Ø1Ø for i=1 to cl
9Ø2Ø print cl$ : rem ezzel mozgatom
                 a kurzort
9Ø3Ø next i
9Ø4Ø return
  .
  .
```

A könyv ajánlja, hogy mindenki használja a gép mellé adott TEST/DEMO lemezen található "C64 WEDGE" segédprogramot.

A gép bekapcsolása után a TEST/DEMO lemezről LOAD "C64 WEDGE", 8 paranccsal töltsük be a segédprogramot, majd RUN paranccsal indítsuk el. Miután a segédprogram bejelentkezett és a képernyőn a READY felirat is megjelent, kivehetjük a meghajtóból a TEST/DEMO lemezt.

A C64 WEDGE segítségével a lemezek kezelése egyszerűbbé válik. Például egy lemezen tárolt PRÓBA1 nevű program így hívható fel a lemezről és így indítható el:

```
↑PRÓBA1
```

Nagyobb programrendszerek pedig például így építhetők fel:

```
1ØØ rem *** — a nagyobb programrendszer
             neve —
    ***
  .
  .
  .
13Ø rem ** — inicializálás — **
  .
  .
  .
  .
  .
  .
5ØØ rem ** — a fő menü
        (a választási lehetőségek) kiírása — **
  .
  .
  .
  .
77Ø input "Melyik lehetőséget
    választja?"; v
775 :: rem a v változóba beolvasom
    a kezelő választását
8ØØ if v=1 then 1ØØØ
81Ø if v=2 then 2ØØØ
  .
  .
  .
1ØØØ rem ** az első választásnak
       megfelelő alprogram hívása — **
  .
  .
  .
1Ø3Ø ↑"alprog1": rem a C64 Wedge segítségével
                 felhívom és elinditom az első
                 alprogramot
  .
  .
  .
  .
2ØØØ rem **— a második választásnak megfelelő alprogram hívása — **
  .
  .
2Ø3Ø ↑"alprog2" : rem felhívom és elindítom a
                  második alprogramot
```

A Commodore 64-est felhasználók nagy része nem szeret ún. random (vagy C64 "tájszólásban": relative) fájlokkal dolgozni, ahol pedig a lemezen tetszés szerinti sorrendben hozzáférhetne a fájl bármelyik rekordjának bármelyik mezőjéhez. A tartózkodás egyik oka minden bizonnyal az, hogy a C64-en a random (relative) fájlok kezelése első látásra kissé nehézkes, a gép mellé adott leírás pedig olyan, mint egy rejtvény.

A C64 random (relative) fájljai egyenlő hosszúságú rekordokból állnak. Minden egyes rekord részekre, ún. mezőkre osztható. A mezők hossza változó lehet. A random fájlt például így lehet megnyitni:

```
  .
  .
14Ø open 2,8,15 : rem a parancscsatorna
                  megnyitása
15Ø open 3,8,4 "pelda1,l,"
               +chr$ (5Ø)
152 rem figyelem! A file neve utáni karakter
  .                kis „l" betű
  .
```

A megnyitott fájl neve "pelda1". A rekordok hossza ebben a példában 5Ø bájt. A parancscsatornához a 14Ø. utasításban a 2-es fájlszámot, a "példa1" random típusú fájlhoz a 15Ø. utasításban a 3-as fájlszámot rendeltük. A további utasításokban a parancscsatornára és a "próba1" fájlra ezzel a két számmal hivatkozunk. A 15Ø. utasítás végén a chr$ (5Ø) adja meg a rekord hosszát, az 5Ø bájtot.

A 15Ø-es sorban érdekesség, hogy a szekvenciális fájlokkal ellentétben nem kell megadni, hogy írásra vagy olvasásra nyitottuk meg a fájlt.

A fájlba úgy írunk, hogy előbb a parancscsatornán keresztül pozicionáljuk a lemezegység (1541) író/olvasó fejét a kívánt rekord kívánt mezője fölé. Így:

```
  .
  .
  .
18Ø print#2, "p"; chr$(4); rec$; mez$
  .
  .
  .
```

A 2-es a parancscsatornára utal. A "p" jelzi, hogy pozicionálunk. A chr$ (4)-ben visszautalunk arra, hogy a 15Ø. utasításban a "próba1" fájlhoz a 4-es csatornát jelöltük ki. A rec$ egy két karakterből álló füzér (string), mely az alább kifejtett, kissé bonyolult módon, 256-os számrendszerben ábrázolva tartalmazza a kiválasztott rekord sorszámát.

A mez$ tartalmazza a rekordon belül azt a bájtot, ahol a kiválasztott mező kezdődik.

Nézzük előbb a rekord megadását:

```
rec$=chr$ (ra)+chr$ (rf)
```

ahol ra és rf egy-egy Ø-tól 255-ig terjedő szám, és a rekord sorszáma, illetve ra és rf között az alábbi összefüggés áll fenn:

```
a kiválasztott rekord sorszáma=256 * rf+ra
```

Megjegyzés: rf — felső bájt; ra — alsó bájt.

A *táblázat* néhány példát mutat a rekord sorszámának megadására a pozicionáló utasításban.

Ha a kiválasztandó rekord sorszáma	akkor	
	ra értéke	*rf* értéke
1	1	Ø
2	2	Ø
255	255	Ø
256	Ø	1
257	1	1
511	255	1
512	Ø	2
513	1	2
.	.	.
.	.	.
.	.	.
.	.	.
stb.		

Az átszámítás a sorszám és *ra*, illetve *rf* között természetesen gépesíthető. Ha a kiválasztandó rekord sorszáma i, akkor ra és rf értékét így számítjuk ki:

```
160 i= ...: rem a rekord
           sorszámának
           megadása
170 gosub 1000: rem ugrás az ra és rf értékét
                kiszámító szubrutinra
    .
    .
    .
    .
    .
    .
180 print#2, "p"; chr$(ra);
             chr$(rf); mez$
    .
    .
    .
    .
1000 rem ** — rekordsorszám
          ki(át)-számítás — **
1010 ra=i
1020 rf=0
1030 if ra>255 then 1050
1040 return
1050 rf=int(ra/256)
1060 ra=ra—256*rf
1070 return
```

A mező kezdő bájtját egyszerűen így jelöljük ki: mez$=chr$ (<a bájt sorszáma>)
Tehát például ha a mező a rekord 25. bájtjánál kezdődik, akkor mez$=chr$ (25).

Ha például a 72. rekord első bájtjától kívánnánk írni vagy olvasni, akkor a pozicionáló utasítás így nézne ki:

```
    .
    .
    .
    .
180 print#2, "p"; chr$(4); chr$(72);
             chr$(0); chr$(1)
```

Ugyanez a 257. rekord 27. mezőjétől:

```
    .
    .
    .
    .
180 print#2, "p"; chr$(4); chr$(1);
             chr$(1); chr$(27)
    .
    .
    .
    .
```

A pozicionálás után az írás már egyszerű. Például:

```
    .
    .
    .
    .
190 print#3, "ez minden"
    .
    .
    .
    .
```

A 180. utasítással pozicionált helyre ezzel fel is írtuk az „ez minden" szöveget. Ha olvasni akartunk volna például az a$ változóba, akkor ezt az alábbi utasítással tehettük volna meg:

```
    .
    .
    .
    .
190 input#3, a$
    .
    .
    .
    .
```

Az alprogram szerkezete olyan lehet, mint a főprogramé. Itt is van inicializálás, itt is lehet menü (almenü). Legyen a főprogram egy leltári nyilvántartó program, melynek egyik alprogramja új tételeket vesz fel a leltárba úgy, hogy előbb kéri az új leltári tárgy nevét, leírását, árát stb., majd a leltárt tartalmazó random fájlhoz fordul, például így:

```
    .
    .
    .
    .
    .
2000 rem ** — alprogram: felírás
             a fájlba — **
2010 ::: rem előbb meg kell tudni, hogy eddig
             hány darab rekord van a
             fájlban. Ezt a számot a program
             célszerűen egy külön szekvenciális fájlban tárolja ugyanazon a
             lemezen. Ezt a fájlt
             pointer-fájlnak hívjuk
2020 : rem * — a pointer-fájl
             olvasása — *
2030 open 2,8,2, "pointer, s, r"
2040 input#2, ptr: rem a ptr változó most azt
             tartalmazza, hogy eddig
             hány rekord van a fájlban
2050 close 2
2060 if status=16 then 3000
2065 :: rem az előző utasítás azért kell, mert ha
         valaki most kezd először dolgozni a
         programmal, akkor még nem talál
         „pointer" nevű szekvenciális fájlt.
         A gép st nevű változója tartalmazza
         automatikusan az erre vonatkozó
         információt. A 3000. utasításnál
         kezdődik az a rutin, mely ebben az
         első esetben létrehozza a "pointer"
         fájlt
2070 ptr=ptr+1
2080 :
2090 : rem * — felírás az adatfájlba — *
2100 open 15, 8, 15 : rem a parancscsatorna
             megnyitása
2110 open 3, 8, 3 "leltár,l,"
             +chr$ (100) : rem
             a rekordok hossza
             100 bájt lesz
2140 ra=ptr : rem itt használjuk fel a szekvenciális fájlból kiolvasott számot
2150 rf=0
2160 if ra>255 then gosub 15000
2165 rem ilyenkor át kell számítani
         256-os számrendszerbe
2170 rec$ =chr$ (ra)+chr$ (rt)
2180 :
2190 mez$=chr$ (1) : rem az első
             bájtból írok
2200 print#15, "p"; chr$ (3); rec$; mez$
2210 print#3, szöveg$ : rem a szöveg $ változó
             tartalmazza azt a szöveget, melyet a lemezre
             fel akarok írni
    .
    .
    .
    .
    .
2380 close 3 : close 15
2390 :
2400 :
2410 rem * — a pointer-fájl tartalmának
             módosítása — *
2420 open 2, 8, 2 "@0: pointer, s, w"
2430 print#2, ptr
2440 close 2
    .
    .
2470 goto 4000 : rem "akarja folytatni"
                 rutin
    .
    .
    .
4080 rem ha nem, akkor ugrás 6000-re, az alprogramot befejező rutinra
    .
    .
    .
6080 főmenü program : rem a főprogram
                 visszahívása
```

Befejezésül nézzük meg, hogy milyen technikát sugall a könyv akkor, ha a kezelőtől jelzést vár arra, hogy tovább lehet-e menni a feldolgozásban:

```
    .
    .
2510 print "Ha folytatni kívánja, nyomja meg a
     RETURN gombot"
2520 gosub 19000 : rem ugrás a RETURN megnyomását figyelő szubrutinra
    .
    .
    .
19000 rem ** — a RETURN
          figyelése — **
19010 poke 198, 0 : rem a klaviatúra-
          puffer törlése
19020 for i=631 to 640
19030 poke i, 0 : rem kiürítés
19040 next i
19050 x=peek (197)
19060 if x=1 then 19080 : rem ha megnyomták
          a RETURN gombot
19070 goto 19050 : rem még nem
          nyomták meg
19080 poke 198, 1 : rem kurzor
19090 poke 631, 0 : rem a klaviatúra
          törlése
    .
    .
    .
19120 return
    .
    .
    .
```

Szóval olvastunk egy jó könyvet. Mindenkinek ajánljuk, aki a C64 fájlkezelésének rejtelmeit meg akarja ismerni. De azoknak is, akik csupán jó BASIC-OLVASÓKÖNYVET keresnek. Ha valamelyik szemfüles kiadónk nem kezdett még hozzá a magyar kiadás előkészítéséhez, akkor már talán késő. Mire a magyar fordítás megjelenne, a könyv valószínűleg aktualitását vesztené. Az is lehetséges, hogy a könyvet szerzője átdolgozza más, újabb gépekre is. Ha ez sem történne meg, annyi haszna lehet ennek a cikknek, hogy közzétette: még BASIC-ben is lehet jó olvasókönyvet írni.

Adjunk ezekből minél többet az angolul nem tudó, vagy idegen nyelvű könyvekhez nehezebben hozzájutó, kispénzű, magyar olvasók (diákok) kezébe is. Nagy szükség van rá. Mert a fiatalok egész nemzedéke fogott neki a nagy számítástechnikai „Egri csillagok", „Csongor és Tünde" megírásához csupán számítástechnikai nyelvtankönyvi ismeretek alapján. Hogyan fog ezekben a művekben például a "tudós monológja" hangzani?

(David Miller: Commodore 64 Data Files, 1984. Prentice-Hall International, Inc., London)

— KE —

FOTO. ELEKTRONIK

Commodore 64-től . . . . . IBM PC AT-ig számítógépek garanciával

Hi-Fi-, video-, fotó- és egyéb műszaki cikkek VÉTELE ÉS ELADÁSA

budapesti és vidéki szaküzleteinkben:

I., Fő u. 37/c	159-869
V., Magyar u. 1.	178-854
V., Múzeum krt. 19.	173-043
V., Váci u. 23.	183-240
VI., Szív u. 38.	
VII., Dohány u. 5.	422-507
VIII., Akácfa u. 59.	222-278
VIII., Baross tér 6.	134-116
VIII., József krt. 40.	131-478
IX., Ráday u. 9.	176-093
XI., Móricz Zsigmond körtér 7.	868-787
XIII., Rajk László u. 46/b	299-604
XIV., Sugár (Örs vezér tere)	836-567
Debrecen, Szabó I. altb. tér 6.	52-29-636
Eger, Széchenyi u. 5.	36-11-649
Győr, Bem tér 1.	96-12-802
Kaposvár, Füredi u. 24.	82-16-307
Miskolc, Korvin Ottó u. 5.	46-17-025
Pápa, Főtér 14.	89-24-402
Pécs, Jókai u. 5.	72-14-302
Székesfehérvár, Széchenyi u. 15/a	22-18-228
Szombathely, Tolbuhin u. 33.	94-18-277

PRONET LOKÁLIS HÁLÓZAT

HARDWARE

— A PROPER—16/A, PROPER—16/W, PROPER—16/m IBM PC kompatíbilis számítógépcsaládból összeállítható hálózat kétirányú busz szervezésű
— A hálózat állomásai közös átviteli közegre kapcsolódnak
— Könnyű a hálózat rekonfigurálása, újabb állomások beiktatása
— A hálózat kiépítéséhez a PROPER—16 számítógépbe bedugaszolható opcionális hálózati kártya szükséges

— Főbb műszaki jellemzők:
- adatátviteli sebesség : 1 Mbit/s
- adatátviteli közeg : csavart érpárú vagy telefonkábel
- a főkábel hossza : maximum 500 m (ajánlott hossz max. 300 m)

— Az állomások elméleti maximális száma 255
(a hálózatba kapcsolt erőforrásoktól és a felhasználás jellegétől függően ajánlott szám kb. 2—10)

SOFTWARE

— PROPOS V. 3. 02 vagy ezzel kompatíbilis operációs rendszer
— Többfeladatos, többfelhasználós környezetben PROMOS operációsrendszer-kiegészítés
A hálózat szolgáltatásainak elérése külső parancsok segítségével, operátori konzolról vagy programból (felhasználói interface) biztosított

AZ Szki STABIL PARTNER!

Számítástechnikai Kutató Intézet és Innovációs Központ
Bővebb felvilágosítás: SCI—L Rendszerértékesítő Iroda
1011 Budapest, Iskola u. 10. Telefon: 153-204

A játékprogramozás technikája

AZ IGAZI JÁTÉK KEZDETE

A zuhanó emberke

A *18. listában* állítjuk be mind a játék kezdetén, mind a játék ismétlődő részében a zuhanó emberke pozícióját. Kezdésnél: MF=0 (280-as sor)

Amennyiben MF=1, akkor a program nem állít elő új zuhanó emberkét.

A helikopter elővétele

A *19. listában* a megfelelő irányú, szembenéző helikoptert kezdő helyzetbe teszi a program *(23. ábra)*. Emlékeztetőül:
CL(23,15)
CR(23,16)
CM(23,17) méretű. A rajz előállításához a következő módosítást tettem:
625 GOTO625

A rajz elkészítésekor a programot „normális" módon futtatjuk.

A helikopter földet érése

A *20. lista* tartalmazza a helikopter földet érése esetén teendőket. Ez egyszerűen azt jelenti, hogy a „földet" újra kell rajzolni. Ugyanez a tennivaló minden olyan esetben, amikor egy tejrelem egy, a továbbiakban ismét szükséges rajzelem helyére kerül, mert ekkor a korábbit letörli.

Tennivalók

A *21. listában* megvizsgáljuk, hogy van-e új zuhanó ember. Amennyiben igen, úgy a program a folytatásra ugrik. Kezdésnél az 580-as sorban levő értéktől függően lehetséges, hogy lesz.

Új zuhanó emberke

A *22. listában* elővesszük a megfelelő pozíciójú emberkét *(24. ábra)*. Emlékeztetőül:
M(5,13)
M1(5,13)
M2(6,13)

A szükséges programmódosítások:
625 nincs
580 MF+1:PM=1
655 GOTO655

A programot itt is „normális" módon futtatjuk.

MELLÉKÁGAK

Tennivalók

A *23. listában* szerepel annak vizsgálata, hogy a zuhanó emberke lezuhant-e vagy sem? Ha igen, akkor a játék „normális" menete megszakad, és a megszakítást vizsgáljuk.

```
570 IFMF=1THEN600 ELSE 580
580 IFRND(NF)<2THEN MF=1
590 MX=RND(220)+12:MY=35
```

18. lista

```
600 IFDI=1THEN PUT(X-3,Y-2)-(X+28,Y+14),CR,PSET
610 IFDI=2THEN PUT(X,Y-2)-(X+22,Y+14),CL,PSET
620 IFDI=3THEN PUT(X,Y-2)-(X+23,Y+14),CM,PSET
```

19. lista

```
630 IFY>125THEN LINE(90,160)-(130,160),PSET
```

20. lista

```
640 IFMF<>1THEN710
```

21. lista

```
650 ON PM GOSUB1410,1450,1430,1450
1410 PUT(MX,MY)-(MX+5,MY+13),M,PSET
1420 RETURN
1430 PUT(MX,MY)-(MX+5,MY+13),M1,PSET
1440 RETURN
1450 PUT(MX,MY)-(MX+6,MY+13),M,PSET
1460 RETURN
```

22. lista

```
660 MY=MY+MS
670 IFMY>147THEN MF=0:GOTO1470
```

23. lista

```
1470 PUT(MX,MY)-(MX+8,MY+15),M,PSET
1480 FORT=3TO18STEP-2:PLAY"V"+STR$(T)+
     "T255O3:DEAD":NEXTT:GOTO1370
```

24. lista

```
1370 PLAY"O1V31T4L2GP100GP80L4GP100L2GL
     4B-AGGGF+G"
```

25. lista

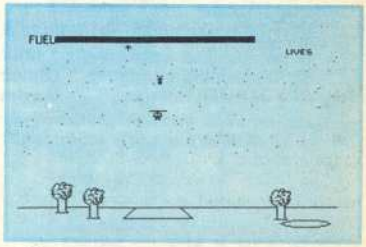
A *24. listában* a földet ért emberkét fölrajzoljuk. Az 1480-as sorban levő furcsa jel, ami már az első cikkben is előfordult, az, amit a „$" helyett a nyomtató kiír. A listában szereplő DEAD szójáték. Angolul ha-

```
1490 FORT=1TO1E3:NEXTT
1500 CLS:A$="GAME OVER"
1510 FORT=1TO9:PRINT@185+T,MID$(A$,T,1);
     :SOUND100-(T*10),1:FORDL=1TO200:NEXTDL,T
1520 PRINT@280,"YOU SCORED";SC;
1530 IFSC>HS THEN HS=SC:GOTO1540 ELSE GOTO1560
1540 PRINT@320,"WELL DONE, YOU'VE BEATEN
     TODAY'S HIGH SCORE!"
1550 GOTO1570
1560 PRINT@320," TODAY'S HIGH SCORE IS";HS
```

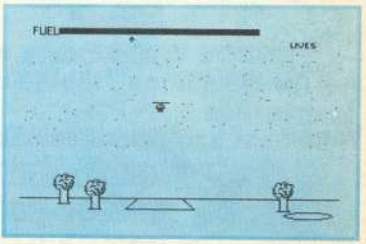
26. lista

```
1570 PRINT@385,"DO YOU WANT ANOTHER GO? (Y/N)"
1580 I$=INKEY$:IFI$="Y" THEN240 ELSE IFI
     $="N" THEN1600
1590 GOTO1580
1600 CLS:PRINT"BYE..THANKS FOR PLAYING!
     ":POKE65494,0:END
1610 FORT=1TO2E3:NEXTT:GOTO220
```

27. lista

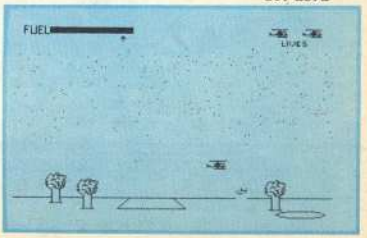


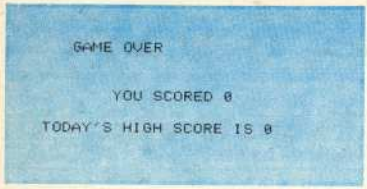
23. ábra



24. ábra

25. ábra





26. ábra

27. ábra

lottat jelent, de egyúttal egy kis dallam is. A kép a *25. ábrán* látható. A szükséges programmódosítások:
655 nincs
1385 GOTO1385
A futtatás itt is a szokásos.

„Életszám"-vizsgálat

A *25. lista* gyászzenét produkál, majd csökkenti az „életek" számát (LI). Ha nincs már több „élet", akkor végeredmény-kijelzés, ha még van, a már ismert eredménykijelzéssel visszatér a játék menetébe.

Végeredmény

A *26. lista* letörli a képernyőt, majd hangaláfestéssel kiírja a végeredményt. (A listában levő „É"-t az „at" karakter helyett írja ki a nyomtató.) Ha az elért pontszám (SC) nagyobb, mint az eddigi rekord (HS), akkor értékelés és ennek kiírása következik *(26. ábra)*.
A programváltoztatás:
1385 nincs
1575 GOTO1575
A futtatás:
RUN 1490

Vége?

A *27. lista* megkérdezi, hogy akar-e még játszani? Ha nem, a program búcsúzik *(27. ábra)*, ha igen, akkor visszatér a játék elejére (a játékszabály ismertetését kivéve).
A módosítások:
1575 nincs
1605 GOTO1605
A futtatás:
RUN 1600

A cikkben szereplő egyetlen új utasítás (26. lista) a SOUND. Ezt két paraméter követi, vesszővel elválasztva. Az első a zörej hangmagasságát adja, és értéke a zongora középső C-jének (1—255)/89 része. A második a hangzás hossza, melynek értéke (1—255)/16 másodperc.

GALINA FERENC

A kritika kritikája

A Mikroszámítógép Magazin 1985/2. számában Várgedő Tamás tolláből bírálatot közölt „A számítógépről mindenkinek" című, Szluka Emil kollégámmal közösen írott könyvünkről (Hírlapkiadó, 1984). Ezt a bírálatot sajnos nem hagyhatom szó nélkül.

A könyvet, amely voltaképpen egy, a Népszabadságban 1984 tavaszán megjelent cikksorozat kissé átdolgozott változata, ketten írtuk. A két rész élesen elválik egymástól: az első részt Szluka Emil írta, a másodikat én. Ilyenformán az első részre vonatkozó bírálattal kapcsolatban illetéktelennek érzem magam, ehhez nem is szólok hozzá, mert nem szeretnék a fogadatlan prókátor nevetséges pózában tetszelegni: ha Szluka Emilnek van mondanivalója, majd elmondja ő maga.

Az általam írott második részről a bíráló csak röviden emlékezik meg. Kissé meglepő számomra, hogy éppen egy számítástechnikai szakember kezelje ironikusan a számítógépet, és írja azt, hogy szerinte hibásan tüntetem fel a számítógépet „a bölcsek kövé"-nek; ezt majd a jövő eldönti.

Ami azonban arra kényszerít, hogy hozzászóljak a bírálathoz, az V. T.-nak az a megjegyzése, hogy „a szerző csak akkor árulja el, hogy *teljesen tájékozatlan* (!— P.G.P.) a témában, amikor ilyen mondatokat ír le" — majd idéz egy mondatot, *egyetlen egyet* a mintegy negyven nyomtatott oldalnyi szövegből. Erre a mondatra mindjárt visszatérek, különös tekintettel arra, hogy vajon ez a mondat valóban „teljes tájékozatlanság"-ról árulkodik-e. Előbb azonban hadd tegyek egy megjegyzést.

Nem vagyok számítástechnikai szakember, sőha nem is próbáltam ilyen színben feltüntetni magamat, jóllehet az elmúlt években sok, a számítástechnikával kapcsolatos cikket szerkesztettem, gondoztam (már az Élet és Tudománynál 1961 és 1968 között is), és nem kevés cikket írtam. E munkám során sok kiváló szakemberrel sikerült kapcsolatba kerülnöm, akik türelmesen oktatgattak a számítástechnikával kapcsolatos ismeretekre, és én igyekeztem figyelmes, jó tanuló lenni. Az első „leckéket" Tarján Rezsőtől kaptam, de sok más „tanáromat" is megnevezhetném, anélkül, hogy bármelyiküknek oka volna szégyenkeznie miattam. Kézirataim elkészítése során mindig igyekeztem a *legjobb szakember segítségét* igénybe venni, nemcsak az informálódásban, hanem *lektoráltattam is* kéziratimat. Bizonyára ennek is köszönhetem, hogy például néhány évvel ezelőtt a Neumann János Számítógéptudományi Társaság sajtópályázatán első díjat nyertem.

Túlzás és hiúság nélkül állíthatom tehát, hogy „teljes tájékozatlanságot" azért a számítástechnika terén aligha árulhatok el. Ezt már csak azért sem tehettem volna meg, mert könyvem — és előtte cikksorozatom — *szigorú szakmai ellenőre* nem engedte volna meg. Nevét fölösleges ideírnom: megtalálja az olvasó a folyóirat impresszumában mint a szerkesztőbizottság vezetőjét.

Nem azért említettem ezt, mert rá akarom hárítani a hibát, hanem azért, hogy figyelmeztessem a bírálót, hogy mintha — számomra ismeretlen okokból — kissé elragadtatta volna magát és elrugaszkodott volna a tényektől.

És ha tényekről van szó, térjünk vissza arra az egyetlen mondatra, amellyel V. T. az én „teljes tájékozatlanságomat" bizonyítani óhajtotta. Ez a mondat a *bírálatban* így hangzik: „A gép memóriája egy megabyte, ami virtuális tárakkal 4 Mbra bővíthető: a tárak mágneslemezek és mágnesszalagok." És ehhez zárójelben még gúnyosan és latin tudását fitogtatandó hozzáfűzi: Sic! Vagyis: így! Nos: *nem egészen így.*

A könyv 102. lapján, a második hasában egy, az IBM 370-et követő sorozatból való gépről ez áll: „A gép *központi* memóriája egy megabyte *(vagy ahogyan ejtjük s „magyarosan" írjuk: megabájt, röviditve Mb= egymillió bájt,* ami virtuális tárakkal 4 Mb-ra bővíthető: a tárak mágneslemezek *(diszkek)* és mágnesszalagok." A két mondat nem egészen azonos: a könyvből kiírt (utóbbi) szövegben feltűnő, hogy a *vagy* szó előtt megnyitott zárójel nem zárul le, ami arra vall, hogy ebbe a mondatba beletenyerelt „a szedőszekrény ördöge", ahogyan az újságírók és a nyomdászok a sajtóhibát nevezik. De ettől függetlenül: *miért bizonyítja* ez az — akár a valódi, akár a bíráló által „idézett" — mondat a szerző „teljes tájékozatlanságát" a számítástechnikában? *Ezt* kellett volna odaírnia a „Sic!" helyett a bírálónak: akkor az olvasó és a megbírált szerző egyaránt okult volna írásából. Ha ugyanis a bíráló csak arra gondol, hogy a *virtuális* szót meg kellett volna ismételni a *tárak* szó előtt, akkor nagyon lebecsüli az olvasók intelligenciáját: ők ugyanis bizonyára megértették, hogy *azok* diszkek és mágnesszalagok. Ha pedig *ez* tévedés, a bíráló feladata lett volna tájékoztatni a szegény félrevezetett olvasót és a tudatlan szerzőt. (A lektorról nem is szólva.)

Várgedő Tamás láthatólag élvezi saját iróniáját, mert bírálatát azzal zárja, hogy könyvünk tanulsága az: a rossz információ, ha értéknek nem is érték, árunak lehet áru. Csakhogy ez nem tartozik a bírálóra — aminthogy a szerzőre sem: ez *a kiadó* dolga. Hogy neki ez az áru *hasznot* hozott-e vagy sem, a könyv *tartalmának* megítélése szempontjából közömbös. Sőt, meglehetősen rosszízű megállapítás. A bírálatnak — legalábbis egy ilyen igényes folyóiratban — meg kellene maradnia a tények szilárd talaján.

PETŐ GÁBOR PÁL

Tisztelt Szerkesztőség!

Furcsa írást olvastam a µMagazin 1986. 3. (márciusi) számában: Simonovits professzor „Számítástechnika" c. tankönyvéről egy eléggé elmaraszta1ó kritika jelent meg. Magával a recenzióval nem kívánok foglalkozni, mert úgy érzem, P. Cs. füstölgését igazából senki sem veszi komolyan — még talán ő maga sem. A vélemény nevetségességén túl, sokkal általánosabb dolgok érdekelnének, nevezetesen az, hogy milyen mértékben tudott a szerkesztőség az írásról, illetve mennyire értettek/értenek egyet vele. Úgy érzem ugyanis, hogy bárki bármit, bárhol elmondhat, leírhat — ez szíve joga. Ugyanakkor egy olyan kérdésben, mint a számítástechnika közoktatása, az Önökéhez hasonló laptól jobb kritikai érzék várható el, mint amiről az írás tanúskodik. Például, vajon miért nem olvashatunk egyetlen vitató sort sem a TV-BASIC c. műsorral/könyvvel vagy az iskola-számítógép rovatukkal kapcsolatban a µMagazin hasábjain? Ezek a szerkesztőség minden tagjának maradéktalanul tetszettek?

A µ '86 alkalmával rendezett szerkesztő—olvasó találkozón többen kifogásoltuk az „Alapozás" címen folytatott tanfolyamot (szerző ugyancsak P. Cs.). Ez az a pont, ahol ismét az Önök véleményére lennék kíváncsi, immáron tehát részletezve a magam kifogásait:

1. Szerintem különös, sőt talán kissé erkölcstelen is, hogy olyasvalaki írjon *ennyire szubjektív,* ledorongoló kritikát egy könyvről, aki maga is

próbálkozik hasonló dologgal. Persze, ezt önmagával kapcsolatban nem mindenki veszi észre, de a szerkesztőség *egyik* feladata éppen ez a kontroll lenne.

2. Pedagógiai szempontból nem lehet mellékes, hogy valami érthető-e vagy sem. A megbírált könyvben — jóllehet — sok apróság vitatható, korrigálandó, az azonban TÉNY — tanárok és diákok tömege bizonyítja —, hogy ÉRTHETŐ, sőt még érdekesnek is mondható. Ugyanakkor hiába kérdeztem végig kb. 100 embert — köztük pedagógust, középiskolást, egyetemistát, szakembert stb. —, egyetlenegy sem akadt, aki véleményt tudott volna mondani az „Alapozás"-ról. Egyikűket sem érdekelte, és egyikük sem értette!

3. Átgondolták-e Önök a szóban forgó recenzió várható hatását tanár—szülő—diák körben (pardon, nem szabályos háromszögben)? Oktatási rendszerünk joggal áll az elmaradtságra kész kritika reflektorfényében, de erről nem tehet sem a számítástechnika, sem egyetlen pedagógus. Ismét kérdem, helyes-e pont őket bántani, ráadásul — szerintem — indokolatlanul?

4. Kíváncsi vagyok továbbá, hogy Önök szerint kinek (milyen rétegnek) az érdeklődésére tarthat számot P. Cs. sorozata, melyhez szinte tökéletesen illeszkedik a 86/3. számban megjelent (induló!?) „OKTATÁS-MÓDSZERTAN"?

Mennyiben azonosulnak a „Zöldségeskert" pökhendi stílusával? Remélem, ezeket csak apró figyelmetlenségek éltetik, avagy Önök is vállalják ezt a tudományoskodó, misztifikáló, szőrszálhasogató stílust, nyelvi csűrés-csavarást, amely eleve lemondhat a megértésről, és teljesen idegen a megkedveltetéstől? Ez jelentené az „informatika társadalmasítását"-t? Azt még megértem, hogy P. Cs. szerint rajta kívül senki másnak nem lenne szabad tollat fognia, de úgy tűnik, a lap is hajlik efelé: a két utolsó számban P. Cs. összesen HAT írással szerepel!

Fenti észrevételekre, kérdésekre nagyon kíváncsian várom a választ. Elismerve a µMagazin sok értékét (a legutóbbi számból pl. a 65XX összefoglaló, több programötlet stb.), nagyon kevesleem a tanárképzéssel, középhaladó-alapozással, ismeretterjesztéssel kapcsolatos *hasznos* anyagokat. A teljesség kedvéért: hiánypótló a „Számítógép anatómiája", általában tetszik a „Favágás", hasznos a „Strukturált programozás". Több számra visszamenőleg azonban korántsem ilyen rózsás a kép. Ugyanakkor úgy érzem, hogy az ország immáron legterjedelmesebb számítástechnikai lapjának nem lehet lényegtelen terület sem a tanárok, sem a haladóbb amatőrök képzése.

Tisztelettel: TÖRÖK TURUL
matematikus
MTA KFKI

Mottó: Fehérek közt egy európai
J. A. Th. Mann üdvözlése

Tisztelt főszerkesztő, kedves Kovács Győző!

Azért fordulok levélben Magához, mert a szó elszáll. A „bűntett az ifjúság ellen" — Pogány Csaba könyvkritikája — a µMagazin ellen is elkövetett bűntény.

Több oknál fogva.

1. A könyv jó.
2. Jobb, mint a szokásos tankönyvek.
3. Ha rosszabb volna, akkor is körültekintőbben kellett volna könyvkritikát közölni.
4. Hát még egy tankönyvről!
5. A kritika hamis és ostoba: (A legsúlyosabbnak tartható Weierstrass „hibásan" kimondott tétele számítógépes környezetben helyes így, hiszen egy korlátos intervallumot a gép csakis zártként tud tekinteni. Csakis diszrét értékei lehetnek.)

Mindezt azért tartom fontosnak közölni, mert még jó másfél évvel ezelőtt, amikor még könyv alakban nem volt olvasható, én Pogány Csaba jelenlétében lelkesedtem — ezzel talán irigységet vagy rosszindulatot keltettem? Ugyanakkor ígértem a könyvsorozat vezetőjének — 85 novemberében — egy kritikát, épp erről, de sajnos nem készültem idejében el vele. A kritikának az lett volna a címe (tudom, szereti J. A.-t): **Fehérek közt egy európai.** Ez nem túlzás. Ez egy kivételes műfajú, szemléletű könyv. Én azt remélem, hogy le fogják fordítani néhány nyelvre.

Üdvözlettel egy volt rovatvezetője
VOTISKY ZSUZSA

Margó

A µM soha le nem irt alapokmányában egy olyan passzus biztosan szerepelne, hogy

— a lap demokratikus fórumot biztosít mindenkinek, aki szakmai véleményét ki akarja fejteni, és talán egy kiegészítés is volna, hogy
— a lap biztosítja a bírálathoz és az ellenbírálathoz is a jogot.

Miután mindkét levél burkoltan arra is céloz, hogy mondjuk el a szerkesztőség véleményét is, megteszem. Elmondom a magamét.

1. A Simonovits-könyvet még nem olvastam, így véleményem csak objektív lehet. El fogom viszont olvasni, mint azt valószínűleg sokan megteszik, akik egyébként ma még nem olvasták el, de P. Cs. kritikája és e két levél olvasata után mohón keresni fogják a könyvesboltokban. A kritika így üzleti szempontból föltétlenül hasznos volt. (Valaki — egy híres ember — egyszer azt mondta: „Mindegy, hogy mit, csak írjanak rólam.") Ennyit a recenzió hatásáról.
2. Lapunk borzasztóan unalmas volna, ha csak azok a cikkek jelennének meg benne, melyek a felelős szerkesztőnek is *és* a szerkesztőbizottság vezetőjének is tetszenek. Ilyen alapú szelekcióról még nem tudok.
3. A lapban megjelenő valamennyi cikkel nem értek egyet, mégis megjelenik. Példaként említeném Pető Gábor Pál A számítógépről mindenkinek c. könyvének kritikáját, amivel azért sem értettem egyet, mert a könyvet én lektoráltam. Tehát tetszett a könyv, mert közérthetően szólt a számítástechnika alkalmazásáról az informatikában képzetlen közönségnek. Úgy látszik nekik is tetszett, mert a könyv elfogyott. Kevésre becsülném magamat, ha a kritika megjelenését megakadályoztam volna csak azért, mert rólam is szól — egyébként okultam belőle.
4. Az „Alapozás" sorozat befejeződött, az olvasók közül az ifjúságnak nem tetszett, az érettebb korosztálytól kapott levelek egy része dicsérte. (Kinek a Papné . . .) A Zöldségeskert stílusát is megváltoztattuk, szimpatikusabb lesz, ha hangvételében a Ludas Matyi „Tücsök és bogár" rovatához fog hasonlítani.
5. A stílusra valóban ügyelni fogunk, már csak azért is, mert bizonyos stílus hasonló stílusú válaszokat indikál, akkor pedig hová jut ez a szegény µM?!

A leveleket megköszönve szívélyes üdvözletét küldi
KOVÁCS GYŐZŐ

PS. A meg sem indult vitát ezzel az írással rögtön le is zárjuk.

György Zoltán, Fehérgyarmat,

Tolnai út 13. 4900

Két éve foglalkozom programozással, BASIC-ben és gépi kódban. Már mind a kettő elég jól megy, most a számítógépek hardverfelépítése foglalkoztat. Főleg a Z80 µP-os gépek érdekelnek, ezek közül is a HT, Primo és a Videoton TV-számítógép.

A kérésem az lenne, hogy küldjék el nekem ezek közül azokat a rajzokat, amik önöknek megvannak.

Aziránt is szeretnék érdeklődni, hogy tervezik-e Z80 µP-os számítógép KIT forgalmazását, s ha igen, akkor mikortól várható ez, milyen áron, felépítésben?

Azt hiszem, hogy a kért rajzokat gépkönyvekben találja meg, ezeket a gyártók az eladott géphez mellékelik, érdeklődésre általában nem küldik el. Mi sem tudjuk odaadni. KIT-ügyben a helyzet változatlan. Nem üzlet, ezért nem gyártják.

András Zoltán, Budapest,

Tito u. 6. IV. 22. 1045

Előljáróban szeretném leszögezni, hogy a hobbiszámítógépekkel és programozással foglalkozó lapok között (Mikro Magazin, BIT-LET, Mikrovilág, Rádiótechnika) én az önök magazinját és a BIT-LET-et kedvelem a legjobban. Erkölcstelennek tartom azt a megoldást, amikor a lap második száma közel a duplájába kerül az elsőnek, ráadásul hosszasan taglalja, hogy miért ne öntsünk forró kávét a lemezegységbe, meg egyéb „hasznos tanácsok" (Mikrovilág).

Szintén nem nyerte meg tetszésemet az a megoldás, amikor beharangoztak érdekes cikkeket (EPROM-programozó, külső programtár stb.), majd a beharangozó utáni számokban már csak azt lehet megtudni, hogy ezek a készülékek megvásárolhatók a DIGITÁL üzletben (Rádiótechnika).

Ezekhez hasonló durva hibákat a magazinban még nem találtam, de ennek ellenére az a véleményem, hogy lehetne jobb is.

Fantasztikus, hogy egy számítógépekkel foglalkozó lap (amely ily módon a csúcstechnológiát képviseli, még ha közvetve is) képtelen elérni azt, hogy a mérete egyforma legyen. A birtokomban lévő 14 szám 4-féle magassággal jelent meg. Ez már csak azért is bosszantó, mert a papírminőség és a külalak egyébként arra inspirálna, hogy beköttessem.

Szívesen látnék több programot. Az a menynyiség, amit C64-re közölnek, megfelelő, de miért csak COMMODORE? Meggyőződésem, hogy a SINCLAIR, PRIMO stb. gépek tulajdonosai is szívesen látnának több, lehetőleg gépi kódot tartalmazó programot.

Nagyon hasznosnak tartottam a repülésszimulációs programokat ismertető cikket. Közismert, hogy a programok itthon „gépről gépre" (kalózmásolat) terjednek, aminek etikusságáról lehet vitatkozni, de ez a valóság. Ennek az a következménye, hogy rengeteg játék, felhasználói program szinte használhatatlan, mert nincs hozzá az alapvető tudnivalókat, információkat tartalmazó leírás. És itt nemcsak a játékprogramokra gondolok! Sikerült megszereznem például a

Amikor megkapom a leveleket, nagyon sokszor úgy érzem, hogy a rovatot át kellene adni már valaki másnak, hiszen nagyon sok szombat-vasárnapom kizárólagos programja a levélolvasás és a válaszírás. Aztán elkezdem olvasni a leveleket és a bosszúság helyét átveszi az öröm. Öröm látni, hogy — elnézést kérek — olvasóink is fejlődnek. Nem kevés kritikát kapunk, amelyekből süt a lap iránti aggódás, a szeretet és a felelősségérzet. Nemrégen valaki bírált néhány írást, talán észre sem vette, hogy úgy fogalmazott: „... a mi lapunk nem engedheti meg magának, hogy ..." Nagyon jó lenne, ha a µM minél többeknek lenne „a mi lapunk", a szerkesztőség mindent megtesz ennek érdekében.

Nagyon sok olvasónktól kaptunk levelet a 85/6. számban megjelent UNIN rutinnal kapcsolatban. Sajnos a nyomtatott lista utolsó oszlopa lemaradt, a javítást a májusi számban közöltük.

Spectrumon futtatható C64-szimulátort, de a mai napig nem tudom felhasználni, mert van valami „trükkje". Egyszóval szívesen látnám a gyári programokat ismertető cikkeket. Gondolom, más is.

Levelére már válaszoltam, itt a levél egy részét közlöm. Ami a lap méretét illeti, az a nyomdai adottságoktól is függ. Ebben az évben az új nyomda a lapot ebben a méretben vállalta. Hogy milyen programokat közlünk, az attól függ, hogy milyen programokat kapunk. Azokat a programokat közöljük, amelyek tartalmilag, programtechnikailag is a legjobbak. Távol áll tőlünk, hogy a Commodore 64-et bármilyen módon is előnyben részesítsük a többi géppel szemben.

A µM-ban gyári programokat csak akkor találj, ha ennek a jogát hivatalosan is megkapjuk, kalózmásolatokkal mi sem foglalkozunk. Nem tagadom, a jó, itthon készült programokat előnyben részesítjük.

Szupper György, Tamási,

Szabadság u. 26/A 7090

Nagy szeretettel olvasom az önök lapját, s mondhatom, igen tetszik. A nevem Szupper György, és nyolcadikos tanuló vagyok. Ne haragudjanak, hogy levelemmel zavarom önöket. Hiányolom a lapból a Commodore 64-re a programokat. Ha lehetne kérni, szeretném, hogy a C64-nek szoritsanak egy kicsivel több helyet. Továbbá még meg szeretném tudni, hogy hol kaphatnék *postán* C64-re játékprogramokat? Az újságot rendszeresen olvasom, és onnan jött az ötlet, hogy önöktől tudakozódjam. Már két helyre írtam, de ott elutasítottak (udvariasan).

Nem tudom, kinek higgyek, Andrási Zoltánnak vagy önnek, tehát sok-e a C64-program, vagy kevés. Maradunk annál, amit A. Z.-nek írtam. Ha megtudom, hogy hol lehet postán C64-játékprogramot kapni, megírom. Addig is forduljon a Novotrade-hez, 1136 Budapest, Fürst Sándor utca 24—26.

Dudás Ernő, Mezőtúr,

Petőfi tér 1. 5400

Szeretném segítségüket, ill. tanácsukat kérni. A gondom az, hogy vásároltam egy CASIO Pb—200 mikroszámítógépet, de nem tudom kihasználni összes lehetőségeit. Ezt értem szoftver és hardver terén is.

Ha megoldható lenne, hogy más, hasonló gép tulajdonosával kommunikálni tudnék, számomra ez nagy segítség lenne a továbbiakban.

Remélem, hogy olvasóink közül valaki segíteni tud, hogy a CASIO-t minél jobban megismerje.

István Takáts, 3690. Bathurs str. # 102.

Toronto, Ont. M6A 2E7 — Canada —

Egy 8 bites ATARI 800XL-re épített rendszer tulajdonosa vagyok és a „Toronto Atari Federation" nevű Computer Klub tagja; aki szeretne minél többet megtudni a számítástechnikáról szórakozásképpen.

Bár a tudásom e téren enyhén szólva alapfokú, kedvenc időtöltésem, hogy felállítok valamilyen problémát és megpróbálok írni rá egy BASIC programot.

Tavaly nyáron Budapesten járva megláttam lapjuk 2. számát, amit elhoztam magammal, állandóan böngészek. Így született az az ötletem, hogy jó lenne valakivel vagy valamilyen klubbal kapcsolatot tartani.

Elég nagy számú közforgalomban levő szoftverrel rendelkezem (főleg a graphics, sound, alkalmazás és az oktatás területén), melyeket örömmel megosztanék az érdeklődőkkel.

Meggyőződésem, hogy véleményt, ötletet, információt és programokat cserélve hasznos ismereteket tudnánk gyűjteni.

Mindig tudtam, hogy kicsi a világ, lám, már másodszorra hallom, hogy a µM eljutott Kanadába is. Remélem, hogy az ATARI-tulajdonosok válaszolnak a szíves hívásra. Ha lesz a HCC-ben ATARI-szekció, egészen biztosan szívesen hozunk létre kapcsolatot távolra szakadt hazánkfiával.

Imre Miklós, Budapest,

Tito u. 8. IV. 20. 1045

Önszorgalommal tanultam meg a BASIC-nyelvet. Előbb egy VC 20-assal, majd egy C64-gyel. Tavaly pedig vettem egy SPECTRUM-ot. Nagyon megszerettem, a hanggenerátorát leszámítva. Ettől a csekélységtől eltekintve, véleményem szerint, bármely mikroszámítógéppel felveheti a versenyt.

Most kezdtem el tanulgatni a Z80-as mikroprocesszor gépi kódú programozását. A BASIC után most evvel is szeretnék megismerkedni. Vettem is könyveket, csakhogy ezekben csak töredékek vannak. Lenne egy kérésem a tisztelt szerkesztőséghez! Szeretném felvenni valakivel a kapcsolatot, akinek ilyen anyagok a birtokában vannak (könyvek, fénymásolatok, vagy kéziratok, egészen az alapoktól). Persze megfelelő anyagi szolgáltatás ellenében kellenének ezek a számomra igen fontos anyagok.

Nagyon hálás lennék érte, ha a szerkesztőség segítene problémám megoldásában.

1. Levelére olvasóink közül valaki biztosan válaszolni fog és segít.

2. Azt javaslom, hogy keresse fel a HCC Sinclair-szekció délutáni klubfoglalkozásait, meglátja, minden kérdésére választ fog kapni. A klubfoglalkozások időpontja: HÉTFŐ 18^{00}—20^{00}-ig. SZELLŐZŐ MŰVEK Bp. XI., ÉPÍTÉSZ U.

Kanyó József, Budapest,

Peterdy u. 39. fsz. 4. 1071

Szüleimtől kaptam egy Commodore 116-os személyi számítógépet. Már nagyon régóta megveszem az önök újságját, de még sohasem találtam benne Commodore 116-os, illetve 16-os játékprogramot. Sőt egyáltalán semmilyen programot C16-os géphez.

Szeretném, ha ezentúl közölnének néhány programot újságjukban C16-os számítógéphez is.

Nagyon örülnék bármilyen programnak.

Előre is köszönöm.

Én nem nagyon szimpatizálok a C16-os géppel, de biztosíthatom, hogy nem ez az oka annak, hogy nem találj ilyen programokat a lapban. Nem küldenek. Ha kapunk jó és érdekes programokat, feltétlenül közöljük.

Szabó Róbert, Makó,

Beloiannisz u. 128/A 6900

Számítógépes szakkörünknek C16-osai vannak, és alig van hozzá „olyan igazi" grafikai játék. Ezért szeretném, hogyha küldenének vagy lapjukban közölnének ilyen programot.

Kérem, olvassa el a fenti válaszomat.

Ziegler Gábor, Baja

Mártonszállási u. 66. 6503

17 éves vagyok, lapjuk régi olvasója. A bajai III. Béla Gimnázium 3. osztályába járok. A mostani számban olvastam a FORTH rendszerekről szóló rovatban, hogy a HT—1080Z gépen futó tinyFORTH programhoz nem találtak magyar nyelvű leírást. Nemrégen történt, hogy kölcsönkaptam róla két angol nyelvű tájékoztatót. (EASY SCRIPT adatállományként ma is megvan.) Címük: TinyFORTH és The tinyFORTH—EDITOR. A gimnáziumban angolul tanulok, te-

hát magamnak nincs szükségem a fordításra. Ha érdekli önöket, szívesen lefordítom és elküldöm közlésre. Tartozott még hozzá egy oldal sűrűn és áttekinthetetlenül gépelt magyar tájékoztató és példák. Ha kell, ezt is elküldhetem, de ez sajtóképes formában több lesz, mint két oldal. (Az angol nyelvű szöveg eredeti állapotban 12 oldal volt, kicsit szellősebben és EASY SCRIPT-tel kinyomtatva jó kétszeresére rúgott.) Várom válaszukat!

Más. Többször olvastam, hogy programok illetéktelen sokszorosítása, árusítása törvénytelen. Engem az érdekel, ha egy programot egyszer, nem üzleti célból, hanem saját használatra lemásolom, megbüntethetnek-e? Szerintem nem, mert ez épp olyan, mintha barátom frissen vásárolt lemezét átmásolom a saját kazettámra. Pedig a lemezre is érvényes a szerzői jogdíj, mégsem hallottam arról, hogy ezért valakit is megbüntettek volna. Ha pénzért teszik másnak, az természetesen más téma, de épp ezért hangsúlyoztam, hogy ingyen és saját használatra teszem. Kérem, válaszoljanak erre is.

Ha a fordítást elküldi, megköszönjük, valószínűleg közölni nem fogjuk, ti. nagyon sok Forth-könyv kapható, ezek számát nem szeretnénk szaporítani. Sorozatot kezdünk viszont Forth-mesteriskola címmel.

Nem vagyok igazán jogász, ezért említett cikkem írása előtt megkérdeztem az ökosokat, hogy szabad-e védelem nélküli kazettát, mágneslemezt átmásolni csak azért, hogy egyvalaki otthon használja, azt mondták „lehet", ti. a copyright csak a terjesztésre vonatkozik, terjeszteni programot engedély nélkül tilos.

Patkós Tamás, 6. oszt. tanuló,

Budapest, Kárpát u. 40. 1133

Olvastam a cikket az újságjukban a „2C" számítástechnikai áruházról. Ebben az állt, hogy az alagsorban lévő gépeken lehet játszani. El is mentem egy barátommal, megkérdeztük, hogy van-e ilyen lehetőség. A válasz: nincs. Az a kérdésem, hogy a cikk írója honnan vette, hogy ebben az áruházban lehet játszani a számítógépekkel?

Nekünk is ezt mondták, a levelet elküldtem a Novotrade igazgatójának, Rényi Gábor elvtársnak.

Somodi Zsolt, Szeged,

Vértói u. 5. 212. 6724

Evvel a levelemmel együtt már a negyedik levelemet írom önökhöz, s közülük csak egyre kaptam választ.

Nem tudom, amit írt nekem, igaz-e: mármint az, hogy megjelent egy könyv a µM jóvoltából, melyben csak programok találhatók (és majdnem mind játékprogram)?

Szeretném megtudni, hogy tényleg megjelent-e, mert én itt Szegeden egyetlenegy üzletben sem találom!

Sajnos, minden levélre nem tudunk válaszolni. Ebben a rovatban azokat az információkat közlöm, amelyekre a válasz talán sokakat érdekel. A könyv megjelent a Műszaki Könyvkiadó gondozásában, a könyvesboltok árulták. A címe: Csupa játék ZX Spectrumra.

Mércsei Kálmán, Salgótarján,

Kemerovó krt. 23. I. 2. 3100

Magamról csak röviden annyit, hogy 32 éves villanyszerelő vagyok. A számítástechnikával kb. fél éve amatőr alapon foglalkozom.

Problémáimat egyedül igyekszem megoldani, segítséget csak végső esetben kérek. Most ezt teszem, mert úgy vélem, nem tudok tovább lépni.

Egy 48 „kilós" PRIMO gépem van. A problémám nem egyedi, azt hiszem ez minden PRIMO-tulajdonosnál jelentkezik, akit ez a gép komolyan érdekel. A BASIC-et különböző módon feldolgozó, ismertető szakkönyvek a rendelkezésemre állnak, de ezek vagy más gépre készültek, vagy általánosságban foglalkoznak a programozással. A gépre szabott könyvem a PRIMO-felhasználói könyvvel kimerül. Az önök lapjának februári számában megjelent rovatszerkesztői megjegyzés 2. pontjában említés történt egy gyártó által megjelentetett hardver és szoftver segédkönyvről. Eddig bárhol érdeklődtem, sehol sem tudnak róla. Kérném, hogy az említett két kiadvány (könyv) megvásárlásához segítsenek hozzá. Ha egy külföldi géptípusról lenne szó, talán magamat okolnám, miért azt a típust választottam, de egy hazai típus esetében . . .?

Levelem második témája az önök lapja. A lapot nem akarom itt méltatni, ezt helyettem mások jogosan megteszik. Csak ennyit: *SIKERLAP.*

Az igények érthetően szerteágazóak. Én, mint PRIMO-tulajdonos, több PRIMO-val foglalkozó cikket, leírást olvasnék szívesen.

A PRIMO-val kapcsolatban azt javaslom, forduljon levélben a gyártóhoz, a leírásról szóló hír onnan származik. A dicséretet szeretjük és köszönjük. Minden jó PRIMO programot szívesen közlünk, ha kapunk.

Tóth Gergely, 7. oszt. tanuló,

Budapest, Abavár u. 45. 1141

Az alábbi, ZX81-re írt program érdekes, kerek grafikus ábrát eredményez. Ha a 2Ø-as sorban levő 44-es számot elkezdjük csökkenteni 5-ig, akkor a „csipkétől" a „napsugárig" különféle érdekes ábrákat kapunk.

```
1Ø FOR N=ØTO 9Ø
15 FOR F=ØTO 18
2Ø LET A=N/44 PI
25 LET K=32+F SIN A
3Ø LET J=22+F COS A
35 PLOT K,J
4Ø NEXT F
45 NEXT N
```

Köszönjük!

Szieberth András, Székesfehérvár,

Sebes u. 17/B 8000

A „Játék a betűkkel, vagy ÁMITÁS" című cikkhez lenne néhány észrevételem.

Az igaz, hogy tökéletesen magyar betűkészletet igen nehéz készíteni, de több olyan program létezik, ami megközeliti azt. Ezek képesek a magas szintű olvashatóságot biztositani, használatuk igen kényelmes, gyors.

A legtöbb szövegszerkesztő program a svéd ábécé betüit használja. Szerintem ez nem akkora fogyatékosság, mint több cikkirojuk szerint. Bizonyitja — bizonyithatja — ezt ez a levél is. Nem hiszem, hogy a tartalom rovására megy a külalak, esetleg félremagyarázhato lenne a szöveg.

A ZX SPECTRUM-ra irt szövegszerkesztők közül az egyik legjobb a TASMURD H, amelyet Vidákovics Attila 1984-ben magyarra leforditott.

Egy korábbi MIKRO-ban volt már róla szó. Használata igen könnyen megtanulhato, felhasználását könnyiti, hogy bármikor lehivhato (természetesen magyarul) a „Manko" — anélkül, hogy a beirt szöveg elveszne. A program az összes létező SPECTRUM-nyomtatoval kompatibilis, mivel ha a program már nem tud mit kezdeni a nyomtatoval, mi is beirhatjuk a nyomtatás-vezérlő karaktereket (persze, ha ismerjük őket).

Én nem rajongok a svéd betűkészletért, zavar, hogy nincs benne i, ó, ő, ú, kellemetlen olvasni pl. a levelét is, amelynek utóirata: „A helyesirási hibákért elnézést kérek!" Tudjon az a gép magyarul, ha már ide került!

Ennyi volt a válogatás az utolsó két hónapi postából. A leveleket szíves örömmel várja:

KOVÁCS GYŐZŐ

Jelentkezem!

Alulirott Endrei Simony rövidesen beküldöm a pályázatra (Mikroszámitógép Magazin 1986. márc. 36. old. — A szerk.) az utóbbi hónapokban végzett munkám eredményét. Csatolom mellé a 200 Ft-ot is, aminek fejében a legalázatosabban kérem olvassa el a mellékelt tájékoztatómat, majd a programot tegye a polcra a többi mellé lebonyolitás végett!

További 300 Ft-ot küldök azért, hogy szíveskedjen lehúzni az ismertető alá odabiggyesztett nevemet, amivel hivatkodóan saját munkámat merészeltem magaménak elismerni. Kérem továbbá, kegyeskedjék a saját nevét, mint jogos tulajdonosét aláírni.

Felhivását kiegészitendő, beküldök még további 400 Ft-ot az újsághirdetés feladási költségeire, amivel majd saját használra vevőket keres a programra.

Kérése szerint küldök továbbá 50 Ft-ot, valamint egy megcímzett, felülbélyegzett és lezárt boritékot feladás végett, amelyben saját magamat értesitem, hogy öt kedves rokona a Jólátvertükőket családból megkapta megérdemelt munkája jutalmát.

Drága nagylelkű jótevőm, maradok örök hive:

Endrei Simony

Alábbi írásunk szerzője beleszeretett a számítástechnikába, és úgy gondolta, hogy szerzett tudásával megpróbálja a helyi iskolát segíteni. Egyszer már levelet váltottunk, akkor biztattam a szerzőt, hiszen törekvése nagyon is egyezik a művelődési kormányzat célkitűzéseivel. Szeretnénk ugyanis, ha az iskolákban nemcsak tanításra használnák a gépeket, de délutánonként pl. klub formában ezek a gépek a felnőttoktatást, sőt a közművelődést is szolgálnák. Különben is az iskolákban ma még nincs elég pedagógus, aki a számítógépekért felelkesedne, és így képes volna a gyerekekkel is megszerettetni a számítástechnikát.
A cikkben a város nevét megváltoztattam, a szerzőét pedig elhallgatom, hiszen a jelenség az érdekes, nem pedig az, hogy hol is történt az eset. Azon sem csodálkoznék persze, ha többen magukra ismernének és végül is belátnák, hogy nagy céljainkat kicsinyességgel, bürokratikus úton nem lehet elérni, együtt, társadalmi összefogással viszont biztosan. Ebben bízik: Kovács Győző.

Számítógépek Kislaposon

Kislaposon a decemberben történt gépvásárlás (2 db C16-os) után jó ideig felbontatlanul, elzárva árválkodtak a masinák. Az indok: nincs hozzájuk színes tévé. Miután azonban sikerült ennek beszerzése is (sajnos csak méregdrága, 26 000 forintos készülék volt kapható), azt gondoltam, hogy elhárultak az akadályok. Felkerestem az iskola igazgatóját, felajánlva eddig „összeszedett" BASIC-tudásomat, könyveimet, például dr. Úry könyvét, a már meglevő és a folyamatosan megjelenő Ötlet és Mikromagazin példányaimat, jelezve, hogy a számítógépek használatához szerintem minderre feltétlenül szükség van. Kértem, hogy rendeljék meg a folyóiratokat és könyveket, vegyenek kazettákat, azzal ugyanis, hogy megvannak a gépek, nem ért véget a munka, hanem éppen hogy el kellene kezdeni.

Mindezt abban a reményben tettem (hiú remény volt), hogy használni akarják a gépeket, és szívesen fogadják tapasztalataimat. Felajánlottam, hogy a levelezést és a jól bevált kazettás programcserét is vállalom, lévén benne a Primo kapcsán egyéves gyakorlatom. Ez a módszer egyben költségkímélő is, azon túl, hogy olyan programokhoz is hozzájuthat így az ember, amelyek nincsenek is kereskedelmi forgalomban, vagy ha kaphatók, meglehetősen drágák. A folyóiratokban mindig találni olyan címeket, ahonnan információk és programok szerezhetők be, ráadásul *ellenszolgáltatás nélkül*, ha nem számítjuk annak a csereprogramokat.

Felajánlottam továbbá segítségemet és szabadidőmet az iskola igazgatójának, és kértem, tegye lehetővé a délutáni órákban, hogy hozzájussak a számítógéphez. Természetesen bárkit szívesen látok, aki érdeklődik, és szívesen dolgoznék együtt a szaktanárokkal. Nem játékalkalmat kértem, hanem lehetőséget arra, hogy lefuttassam az otthon papíron megírt egyszerű BASIC-programjaimat, és a könyvből, amely az iskolának nincs meg, beírhassam és kazettára vehessem a példaprogramokat. Ugyanezt terveztem a folyóiratokban már megjelent programokkal is. Úgy gondoltam, illik ezzel mintegy honorálni a géphez jutás lehetőségét. Azzal is biztattam, hogy valószínűleg előbb-utóbb kereskedelmi forgalomba kerülnek majd szaktárgyi oktatóprogramok is. Ez azóta meg is történt.

Az igazgatót hidegen hagyta a lelkesedésem. Sehogy sem tudta megérteni például, hogy mire is kell az üres kazetta. Az első problémát a számítógéphez való hozzáférés jelentette, ugyanis én csak munkaidő után érek rá, akkor viszont az iskola és a gépek is gondosan be vannak zárva. A géppel való „foglalkozásra" két matematika szakos tanár lett „kijelölve", ők mindketten éppen alapfokú BASIC-tanfolyamra jártak — hol jártak, hol nem — Nagylaposra. Igaz, hogy ott nem találkozhattak C16-ossal, de a programozás alapjaival megismerkedhettek.

Ennek megörültem, legalább lesz kivel együttműködnöm. De örömöm korai volt, mert kiderült, hogy egyikük nemsokára szülni megy, és a másik sem ér igazán rá, mivel kémia-továbbképzésre jár, ami minden energiáját leköti, és különben is kétgyermekes családanya, munkaidő után siet haza. Otthon talán szakíthatna némi időt a géppel való ismerkedésre, de jaj!, a gépet hazavinni, azt nem lehet! Nem baj, javasoltam, addig én magam foglalkozom a géppel, és ha majd a feladattal megbízottak is ráérnek, legfeljebb átadom a tapasztalataimat.

Így sikerült is 2-3 alkalommal délutánonként néhány órát töltenem a géppel négyszemközt. Éppen a kapott játékprogramokat futtattam, de az utolsó kettőre már nem jutott időm, mert a takarítónő is befejezte már a munkáját.

Másnap újra megjelentem, de legnagyobb megdöbbenésemre a szertár kulcsa már nem volt a szokásos helyén, így a takarítónő nem tudta ideadni a gépet. Egyébként bizalmasan közölte velem, hogy többen kifogásolták, miért babrálhat egy idegen a géppel, amikor ők még nem is látták, különben is a két tanár nevén vannak a gépek, és mi lesz, ha elrontom őket.

Nem feszegettem a „kulcskérdést", inkább szóba hoztam, hogy szívesen vállalnám egy szakkör beindítását, mivel a megbízott szaktanár decemberig nem ér rá. — Hát arról szó sem lehet — válaszolták —, mert szakkört csak szeptembertől lehet indítani! Velem lehet alkudni — mondtam —, hívhatjuk másképpen is a foglalkozást. Nem, nem, különben sincs rá keret. — Ejnye — mondtam —, elvállalnám én díjazás nékül is, de ha ez zavarná, a TIT-en keresztül megszervezhetjük, így volt elintézve a Primós tanfolyammal is. — De így sem jó, mert éppen ma hallottam a GAMESZ-től (ez a szerv intézi az iskola pénzügyeit), hogy a szomszédos Alsólaposon Felsőlaposról érkezett szakemberek üzemelték be a számítógépet, és amíg ide is el nem jönnek, inkább ne bántsuk a szegény gépet! Furcsa — mondtam —, ugyan mit kell azon a gépen beüzemelni? A beüzemeléshez szükséges összes tudnivalót tartalmazza a gépkönyv is. — Na nem baj — mondta az igazgató (ez éppen pénteken történt) —, hétfőn majd rákérdezek, azután eldöntjük, mi is a teendő. Én persze nem vártam hétfőig, még aznap megtudtam, hogy a számítógéphez vett színes tévéhez a Gelka tetőantennát szerelt fel, mert a GAMESZ szerint a számítógéphez ez szükséges, különben még elromlik!

Többet nem engedtek a számítógéphez nyúlni, pedig hajlandó lettem volna anyagi felelősségemet írásba is adni. Egyszerűen nem vagyok jogosult a kezelésére, mivel nem tartozom a tanári karhoz. Érdeklődtem, hol van valamilyen tiltó jogszabály. A válasz: ilyenről ugyan nem tudnak, de olyanról sem, ami ezt lehetővé tenné! (Elismerem, remek riposzt.) De egyébként is már pusmognak a kartársak, hogy hogy jön egy idegen ahhoz, hogy az iskola számítógépét babrálja! Egyébként meg — tolták elém a tematikát — mi nagyon jól tudjuk, mire való egy számítógép, maga is jobban teszi, ha elolvassa. Hát elolvastam, hangosan: matematika, valószínűségszámítás, algoritmusok stb. Hoppá — mondom —, itt a lényeg, amiről eddig hiába beszéltem. De pechem volt, mert addig már nem jutottak el az olvasásban.

Ekkor értettem meg, miért ragaszkodnak foggal-körömmel ahhoz, hogy csak a matematika szakos tanár jogosult a használatra, a többinek meg minek a számítógép. Újra makacsul magyarázni kezdtem, hogy a matematikánál jóval szélesebb körben használható, ha van hozzá megfelelő program. Erre azt vágták a fejemhez, hogy fél, hogy a tanulmányi munka rovására menne a dolog, mert 45 perc amúgy sem elég az oktatásra, hát még ha közben a számítógéppel kell játszadozni!

Szívesen tartanék foglalkozást nyáron is — mondtam —, akkor legalább biztosan nem menne a tanulmányi munka rovására. Az is nagy szó lenne, ha *csak* játszanának gyerekek a géppel. Sőt jöjjön el a szülő is, hadd lássa, hogy végre az ő gyereke is hozzájuthat Kislaposon a számítógéphez. Ez a javaslatom sem tetszett.

Próbáltam fejtegetni, hogy az iskolának nem szabadna ilyen mereven elzárkóznia attól, ami egyébként szerencsére máshol már jól bevált gyakorlat, hogy az intézmények a szabad gépidőt klubformában elérhetővé teszik az érdeklődő fiatalok és felnőttek számára. Azt mertem mondani, hogy szerintem tágabb értelemben a gép tulajdonképpen nem is az iskoláé, hanem a falu közösségéé, lévén a vásárlást eszközlő tanács állítólag a falu gazdája. Na ennél eretnekebb dolgot nem is mondhattam volna, ezzel betelt a pohár. Tudtomra adták, hogy a számítógép igenis az iskola tulajdona, és mint ilyen, kizárólag az iskola rendelkezik vele.

Nem láttam értelmét a további kötélhúzásnak. Annyit még utánam vetettek, hogy majd szólnak, ha szükségük lesz a segítségemre, amit egyébként köszönnek. Én is!

Eddig tart a történet. Azt hiszem, jókora tanulsággal szolgálhat. Ezért is mertem megírni. Egyetlen remény éltet még: hátha egyedi eset az itteni, hátha máshol elő sem fordul hasonló. Tenni kellene annak érdekében, hogy ne is fordulhasson elő ilyesmi, ne értsék félre szándékosan az írott szót, ne a hatalom diadalmaskodjon a jó szándék felett, mert egyébként jaj a haladásnak!

Volt egy olyan érzésem, mintha az egész nem is Magyarországon történne, hanem egy információktól elzárt szigeten, ahol kőkorszaki gondolkodási sémák rögzültek. Elragad a szenvedély: szerintem az ilyen magatartás bűn, és ellentétes azzal az egyre többet hallott jelszóval, hogy „A számítástechnika mindenkié!".

BITEK ÉS FIGURÁK

Állásértékelés III.

Az eddigiekben megismerkedtünk az állás értékelésének az anyag, a mozgékonyság és a centrumellenőrzés komponenseivel. Ez azonban az állás pontos értékeléséhez még nem elégséges, ezért további szempontokat is figyelembe kell venni. Ez alkalommal a fejlődés és a sáncolás jelentőségével, ezek kvantitatív meghatározásával foglalkozunk.

Az értékelőfüggvény fejlődéskomponense

A fejlődés szerepe tulajdonképpen a megnyitás kezdeti stádiumában a leglényegesebb. A sakkjáték megnyitásában fejlődnek ki a figurák. Az a cél, hogy a középjátékban minél aktívabb szerephez jussanak. A játszmában az van kedvezőbb helyzetben, aki tempóelőnynyel rendelkezik, mert tulajdonképpen ez diktálja a játszma további menetét. Ezért nagyon fontos, hogy figuráinkkal minél előbb kifejlődjünk, és ennek érdekében a játszma elején ne lépjünk többször azonos bábuval, hanem próbáljuk meg a teljes haderőt mozgósítani. Sakkoktatásokon már a legalacsonyabb szinten is tanítják az „anyagot áldozok, időt nyerek" elvet, melyben az egyik játékos anyagot áldoz a fejlődés érdekében, hogy tempóelőnyt szerezzen. Ez sok esetben már meghatározza a parti végső kimenetelét.

A fejlődés és a korábbi számokban említett mozgékonysági érték szorosan összefügg. Amelyik félnek fejlettebb a hadereje, annak a mozgékonysági tényezője is nagyobb. A mozgékonysági konstans tárgyalásánál említettük a különböző cseljátékok előnyeit, amelyek azon alapszanak, hogy az egyik fél anyagi áldozatával a figuráit kifejleszti, és ezáltal a bábuk mozgékonysági értékét is növeli. Alapfokon általában azt tanítják, hogy a fejlettségben három lépés előny körülbelül egy gyalogot ér, ami eléggé durva közelítés, de sokszor jól tükrözi az állás aktivitását. Nézzük a következő lépéssorozatot, amelyben egyértelműen látható a sötét haderő fejlődési előnye:

1. Hg1—f3 d7—d5
2. Hf3—g1 Hb8—c6
3. Hg1—f3 Hg8—f6
4. Hf3—g1

Ha két egyforma erős játékos innen folytatná a játszmát, a parti sorsa egyértelműen sötét javára billenne. Ez a példa mesterkélt, de kifejezi a lépésismétlés hátrányát, amely rossz játékvezetéssel a megnyitás későbbi szakaszában gyakran előfordul. Először a könnyű tisztekkel (futóval, huszárral) kell kifejlődni, mert ellenkező esetben — ha a bástyát vagy a vezért centralizáljuk — az ellenfél figurái fejlődés közben tempót nyerhetnek, ha bástyánk és vezérünk megtámadásával újabb bástya- vagy vezérlépésre kényszerítenek.

A fejlettség értékelésében sokféle tényezőt hasznos megvizsgálni.

Először is meg kell vizsgálni, hogy mely figurák hagyták el eredeti helyüket. Ennek figyelembevételével az előbb említettekkel egybehangzóan elsőként a huszárok és a futók kifejlesztése, majd a sáncolás és végül a nehéz tisztek aktivizálása a helyes sorrend.

A fejlettség másik, kevésbé ismert szempontja, hogy ameddig vezér van a táblán, addig a sáncolás majdnem létfontosságú. Ennek több oka van. A legtöbb megnyitásban a középvonalak jóval hamarabb kinyílnak, mint a szélsők. A centrumban bekövetkező cserék során ezek a vonalak sok esetben viszonylag korán szabaddá válnak a bástyák vagy a vezér számára. Így érthető, hogy a középen maradt királyra ezeken a vonalakon több veszély leselkedik, mint a sáncolás után, amikor az eredeti helyükön levő f, g és h gyalogok (vagy hosszú sánc esetén az a, b és c gyalogok) szinte a saját testükkel oltalmazzák a király életét. Az egyensúly megbillenését pedig azáltal okozza a középen rekedt király, hogy megszakítja a két szárny — a vezér- és a királyszárny — közötti összeköttetést. Így szinte közlekedési akadályt képez, és kettéosztja saját haderejét, meggátolva annak összerű elhelyezkedését és átcsoportosítását. Ha viszont a vezérek lecserélődtek — legalábbis az ellenfél vezére nincs a táblán —, akkor már nem olyan alapvető a sáncolás.

Hogyan lehet ezeket a szempontokat beépíteni az értékelőfüggvénybe, hogyan lehet ezeket kvantitatív módon kifejezni? — erre David Levy egyszerű, de érthető példát adott. Ebben a mennyiség ugyan nem pontos, de eléggé érzékeny ahhoz, hogy a gondolatmenetet világosan tükrözze, és ezáltal egy sakkprogram írásához ötletet adjon, amit tovább lehet finomítani. Levy példája:

Fejlettség =
D/3–U/4–(k*C)

ahol:

D = azoknak a könnyűtiszteknek a száma, amelyek nincsenek eredeti helyükön;

U = 0, ha a vezér még nem mozdult, vagy leütötték, illetve
= a fejletlen könnyű- és nehéztiszték száma, ha a vezér már elmozdult, de még nem ütötték le;

C = 2, ha az ellenfél vezére még a táblán van, illetve
= (1—P/4), ha az ellenfél vezére már nincs a táblán, ahol P a kölcsönösen leütött bástyák és könnyűtisztek száma;

k = 0, ha a játékos elsáncolt;
= 1/3, ha a játékos elvesztette a vezérszárny sáncolási jogát;
= 2/3, ha a játékos elvesztette a királyszárny sáncolási jogát;
= 1, ha a játékos mindkét szárnyon való sáncolási jogát elvesztette.

Az így megadott függvény első két tagjában külön ki kell emelni, hogy a "D" változóban nem az eredeti helyükön levő tisztek számát kell figyelembe venni, hogy az eredeti helyen leütött tisztért ne járjon fejlődési büntetés, mivel az a saját hibáján kívül nem léphetett még. Ha a játékos vezére még a táblán van, de már mozdult, akkor a játékos minden tisztjéért büntetés jár a függvény második tagja szerint. A fejlettségi függvénynek ez a két tagja biztosítja a már említett stratégiát, hogy elsőként a könnyűtisztek fejlődjenek ki, majd csak ezután következzen a vezér és a bástyák aktivizálása. Tulajdonképpen ez a két tag áttételesen a sáncolást is elősegíti, mivel ez — mint kettős lépés — a királyt is és egyben a bástyát is kifejleszti. Azonban a tapasztalat szerint a sáncolás ösztönzésére ez kevés, ezért kell külön figyelembe venni a fejlettségi függvény harmadik tagjában a játékos sáncolási jogának elvesztését. A "k" szorzó értéke akkor a legnagyobb, ha a játékos mindkét oldali sáncolási jogot elveszítette. Egy fokkal csekélyebb e szorzó értéke, ha csak a királyszárnyi sáncolás joga veszett el, és még csekélyebb, ha csak a vezérszárnyi. A királyszárnyi sáncolás a gyalogok elhelyezkedése miatt biztonságosabb a vezérszárnyinál.

A sáncolási jog elvesztésének megállapításánál a már említett okokból figyelembe kell venni, hogy az ellenfél vezére fent van-e még a táblán. Ennek megfelelően a "C" szorzó értéke kettő, ha a vezér játszik. Viszont ha az ellenfél vezére elvesztett, vagy lecserélődött, akkor a sáncolás fontossága csökken, és így "C" értéke a következőképpen számítható ki:

1—(P/4), ahol P a kölcsönösen leütött tisztek száma, beleértve mind a könnyű- mind a nehéztiszteket.

Ha jobban megvizsgáljuk ezt a kifejezést, akkor kiderül, hogy "C" értéke negatív is lehet. Ez akkor következik be, ha például az ellenfélnek már csak egy tisztje van, és így végjátékra kerül sor. Ez arra ösztönzi a programot, hogy ebben az esetben mellőzze a sáncolást, mert ez a végjátékban kimondottan hátrányos. Ott ugyanis a királynak aktív szerephez kell jutnia.

Nézzünk példaként egy játszmát, amelyben a mozgékonyság és a fejlettség játssza a vezető szerepet. Ozsváth András és Eperjesi L. 1967-ben játszotta.

1. e4,c6 2. Hc3,d5 3. Hf3,Fg4 4. h3,Ff3: 5. Vf3:, Hf6 6. e5!?,Hfd7 7. Vg3!,e6 8. d4!?,Vb6 9. Fd3!,Vd4: 10. 0—0,g6 11. Fe3,Vb4 12. a3!,Va5 13. b4,Vc7 14. f4,a6 15. Bae1,c5 16. f5!,gf: 17. Ff5:,Hc6 18. Fg5,He7? 19. Fg4,h5 20. Vh4,Hg6 (lásd az *ábrát*).

Láthatjuk, hogy világos figurái mennyivel mozgékonyabbak, fejlettebbek, mint a sötétek. Világosnak minden tisztje mozgott, kifejlődött, és így aktívan részt vesz a küzdelemben. Királyát sáncolással biztonságba helyezte. Sötét viszont csak passzív szerephez jutott, királya a középen rekedt, ennek következtében királyállása is nagyon bizonytalan. Futója és mindkét bástyája az eredeti helyen maradt, és így ezeknek a figuráknak az aktivitásáról nem is beszélhetünk. Világos ki is használja mozgékony állását, és egy leleményes kombinációval — melyet a számítógépek is hamar megtalálnak — gyorsan megnyeri a partit.

21. Hd5:!,Vc6 (Vc6:-ra 22.e6! gyorsan dönt)

Ha most tüzetesebben megvizsgáljuk az állást, és az előbb megadott fejlődési függvénybe behelyettesítjük az értékeket, akkor bizonyságot szerzünk arról, hogy a közelítés eléggé pontos. Nézzük konkrétan a függvény egyes komponenseinek értékét!

D: nem az eredeti helyükön levő könnyűtisztek száma: 3 világos, 2 sötét.

U: világos esetében 0, mivel a vezér már elmozdult eredeti helyéről, viszont minden figurája ki is fejlődött. Sötét számára viszont 3, mert vezérét tisztjeinek kifejlődése előtt aktivizálta; ez büntetőpont levonásával jár.

C: mindkét fél esetében 2, mert a vezérek a táblán vannak.

k: 0 a világos és sötét számára is annak ellenére, hogy sötét nem sáncolhat, mert a világos futó ütésvonalát nem lépheti át. (Ennek a kivételes esetnek a pontozására, értékelésére későbbi cikkünkben visszatérünk.) Így a végső eredmény a következőképpen alakul:

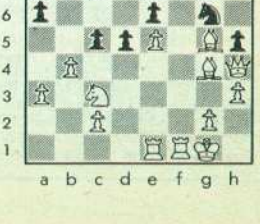
FV = 3/3—0/4—2*0,
FS = 2/3—3/4—2*0

Itt FV a világos, FS a sötét fejlődési függvényt jelöli. Látható, hogy világos pontértéke +1, míg sötété −1/12, tehát a különbséget jól szemlélteti. Ha a mozgékonysági értéket is figyelembe vesszük, akkor még nagyobb különbség látható: világosé 40, míg sötété csak 32. Így világosan kitűnik, hogy ezeknek a szempontoknak a figyelembevételével az állás milyen pontosan értékelhető. Ez a számítási mód finomítható, még hatékonyabbá tehető.

Érdekességként nézzük meg a játszma csattanós befejezését.

22. Fh5:!!,Bh5: 23. Vh5:,Vd5: 24. Bf7:!!,Hde5: 25. Be5:!Vd4+ (25. —,Ve5: 26. Vg6:, vagy 25. —,He5:? 26. Bb7:+ természetesen szintén reménytelen sötét számára) 26. Bf2!, Ve5: 27. Vg6:+,Kd7 28. Vf7+,Kc6 29. Ff4!,Va1+ 30. Kh2 és sötét feladta.

KOVÁCS P. ATTILA



Kész szoftvertermékeket ajánl IBM PC/XT és ezzel kompatíbilis számítógépekre a

SOFTinvest
SZOFTVERKERESKEDELMI ÉS FEJLESZTÉSI BETÉTI TÁRSULÁS

PcFOK
főkönyvi és folyószámla-könyvelési rendszer

Általános jellegű főkönyvi könyvelésre alkalmas programcsomag, amelynél — egy újszerű szervezési megoldás révén — az adatbeviteli munka mintegy 60%-kal csökken, és a havonként elkülönülő zárás lehetővé teszi a folyamatos könyvelést.

PcGAZD
készletgazdálkodási rendszer

A programcsomag naprakész információt szolgáltat a készletek alakulásáról, ezzel egyidejűleg
- főkönyvi feladással támogatja a főkönyvi könyvelést,
- utókalkulációval az anyagellátást,
- anyagfelhasználási mérleg készítésével a statisztikai adatszolgáltatást,
- likvidációs feldolgozásokkal a pénzügyi elszámolást,
- leltárfeldolgozásokkal a leltározási munkát.

PcÁLLÓ
állóeszköz-nyilvántartási és -gazdálkodási rendszer

A programcsomag gondoskodik az állóeszközök értékadataiban bekövetkezett változások átvezetéséről, főkönyvi költséghelyi feladásáról, a mérlegtáblák automatikus kitöltéséről amellett, hogy tartalmazza a statisztikai, műszaki és leltározást támogató információk előállítását.

A mielőbbi közvetlen kapcsolat reményében várjuk megrendelésüket.

Cím: 1024 Budapest, Petrezselyem u. 6.
Telefon: 351-950 és 351-972

ÚJ

SZOFTVERHÁZ 1027 Budapest, Csalogány u. 9—11.
Tel.: 355-521, 352-151, telex: 225138

1986. július 1-jével a számítástechnikai fejlesztések komplexebb ellátása, a piaci igények hatékonyabb kiszolgálása érdekében az ÉGSZI vezetése megalakította a SZOFTVERHÁZ-át.

FŐ MŰKÖDÉSI TERÜLETEK:

— kis- és mikroszámítógépes hálózatokra szoftverfejlesztés (RC 3600, PDP-alapú kisgépek, IBM PC/XT/AT, PROPER—16)
— osztott rendszerek fejlesztése. Referenciahelyek: SKÁLA, FÜSZÉRT, HÍDÉPÍTŐ
— magasraktár-szervezés
— vonalkódtechnika alkalmazása

Részletesebb felvilágosítást a fenti telefonszámon SZULA Szabolcs igazgató, TURAI Tamás igazgatóhelyettes, dr. RÉVFALVY Miklós, marketingvezető ad.

AZ ORSZÁG MINDEN RÉGIÓJÁBAN ÉGSZI

C16 klub

Az elmúlt év végén jelentek meg a hazai boltokban a Commodore 16-os mikrogépek, nálunk akkor még szokatlanul olcsó, nyolcezer forint alatti áron. Tavasszal az általános iskolák számára iskolagépként ajánlott típusok listájára is felvették. Szinte egyik napról a másikra, gyorsan terjedtek el, és ezért kevés hozzájuk a szoftver, a dokumentáció és általában a tapasztalat.

A felhasználók intézményes segítségnyújtására a SZÁMALK KISZ-bizottsága vállalkozott. Létrehoztak egy Commodore 16 klubot. Vettek néhány gépet, ezek képezik a klub alapját. A klubösszejöveteleket hétfőnként tartják a SZÁMALK Szakasits Árpád út 68. szám alatti épületének aulájában.

Segítséget nyújtanak a klubtagoknak a programírásban, sőt a kész programtermékek menedzselésére is vállalkoznak. Megbízást adnak különböző segédprogramok megírására; például készültek már a klubmunka keretében különböző turbo-programok, assembler-segédprogramok, a képernyőtartalmat kinyomtató programok és egyéb ötletek megvalósításai. Ezek közül a piacképeseket az ÁSZ-bolthálózat forgalmazza. A klubfoglalkozások nyilvánosak, se belépődíj, se tagsági igazolvány nincs.

Beszédszintézis házi gépre is

Új magyar szabadalmon alapuló beszélő perifériát készített a BME Híradástechnika Elektronikai Intézete. A készülék a széles körben elterjedt soros adatátviteli vonalon (RS232) keresztül vezérelhető, így minden olyan számítógép, személyi és mikroszámítógép, képernyős megjelenítő, amelynek RS232 kimenete van, beszélhet a segítségével; a ZX-Spectrum például az Interface 1-en, a Commodore 64 pedig egy RS232 illesztőadapteren keresztül. Egy-egy szó vagy kifejezés kimondása egy vezérlő kódszó kiadását igényli, így a központi egységet csak igen rövid időre foglalja le.

A készülék saját memóriájából mintegy 300 magyar szót képes kimondani. Az alapszótáron kívül speciális szókészlet is igényelhető, amelynek kifejlesztésével a BME-n foglalkoznak.

Az új periféria főbb felhasználási területei a parancsok, utasítások, eredmények kimondása, az oktató- és játékprogramokban való alkalmazás, akusztikus jelzések és riasztás a folyamatirányítás és a termelésirányítás terén, valamint a beszélő ipari robotok előállításánál.

A látássérültekért

Tavasszal 23 db hazai gyártmányú Brailab típusú gépet kaptak a látássérültek a Tudományszervezési és Informatikai Intézettől. A Brailab gépet a szocialista országok első, gyártásba vett házi számítógépéből, a Homelabból alakították ki a Lukács testvérek. A gép minden lenyomott billentyűt számítógéppel szintetizált hanggal visszajelez, amit hangszórón vagy fülhallgatón keresztül hallhat a gép kezelője. A számítógépes beszédszintetizálási feladatot a KFKI szakemberei oldották meg.

A Vakok Országos Szövetsége tavasszal az új számítógépek alkalmazásba vételére tízhetes tanfolyamot indított. Ennek feladata egyrészt az általános számítástechnikai kultúra emelése, másrészt a tehetségkutatás, a továbbtanulás előkészítése volt. Ez utóbbi vonatkozásában mind a SZÁMALK, mind az ELTE kész a segítségre. Például a SZÁMALK már az elmúlt ősszel beiskolázott egy vak hallgatót.

A számítógép-alkalmazási program célja, hogy a vakok számára újabb munkahelyeket teremtsen. Szóba jöhetnek például a gépíróiról, a telexkezelőiről, a diszpécser és az adatbázis-kezelői munkakörök.

PRODUKTORG
SZERVEZÉSI VÁLLALAT

C64, M08X, IBM PC XT kompatibilis rendszerekre ajánljuk programcsomagjainkat:

— főkönyvi könyvelés,
— költségfelosztás,
— állóeszköz-nyilvántartás,
— rendelés-nyilvántartás,
— bérelszámolás,
— adóelszámolás.

Bővebb felvilágosítással készséggel állunk rendelkezésükre!

Bővebb felvilágosítással készséggel állunk rendelkezésükre!

Személyiszámítógép-javítás, karbantartás közületeknek, magánszemélyeknek.
Egyedi megrendelés alapján kiegészítő berendezések gyártása.
Pl. Sinclair fényceruza, Joystick Interface, oktatási intézmények kabinet kialakítása.

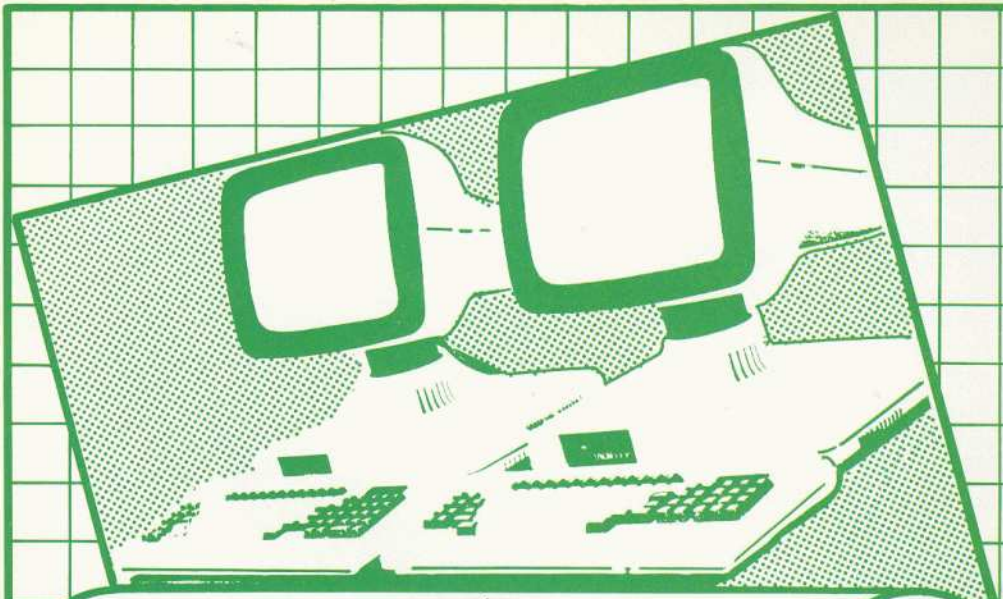
**Pásztor Ferenc személyiszámítógép-javító és -karbantartó kisiparos.
Szolnok, Mátyás király u. 2. V/3.**

SZÁMÍTÓKÖZPONT-VEZETŐK!

Ne dobják ki sérült mágneslemezeiket!
Gyorsan,
olcsón,
6 havi garanciával megjavítjuk!

Commodore 64 számítógépükhöz
— az adatrögzítést megkönnyítő —
kiegészítő numerikus klaviatúrát
raktárról szállítunk,
1 éves garanciával.
Megvásárolható
az Econorg
1. sz. számítástechnikai
szaküzletében:
Bp. VI., Szinyei Merse Pál u. 1.
Tel.: 127-628

**Egészségügyi Elektronikai Gm
1045 Bp., Erzsébet u. 14. X. 59.
Telefonügyelet: Grósz Andor, 632-720 (9—17 óráig)**



KI A LEGFONTOSABB EGY IRODÁBAN?

a VT 16
a titkárnő
titkárnője

Természetesen a titkárnő.
Leveleket megír és átír, címeket tart fejben, nyilvántartja a hivatalos iratokat.
A tipográfiához is értenie kell, hiszen szerkeszteni, tördelni kell a levelek, iratok szövegét.
Elvégzi az ismétlődő szövegrészek és az első példányok újragépelésének munkaigényes feladatát.
Olykor azonban a legtökéletesebb titkárnő is elfárad.
És máris ott a hiba lehetősége.
Hacsak!
Hacsak nincs a keze ügyében a VT 16 számítógép, „akire" nyugodtan rábízhatja magát. Mert

Magyar karakterkészletes billentyűzet és nyomtató, 256 kbájt memória.
Nagy grafikus felbontású, szemkímélő képernyő.
Ára: alapgép floppy diszkkel 260 000,— Ft
Winchester diszkkel 440 000,— Ft
Országos vevőszolgálati hálózat!
Felvilágosítás: 804-133.

A VIDEOTON PROGRAMJA – A JÖVŐ PROGRAMJA

SOFTINVEST
SOFTINVEST
SOFTINVEST
SOFTINVEST
FWINDOW
MicFORTH
dACCESS
MicFORTH
354-350